RAPPORT FINAL.

Projet RNTL : «Nouveaux modèles économiques, nouvelle économie du logiciel».

Coordonné par Nicolas JULLIEN, Mélanie CLÉMENT-FONTAINE, Jean-Michel DALLE.

Réalisé grâce au soutient du 

Table des matières

Liste des tableaux

Liste des figures

Introduction générale.

NICOLAS JULLIEN, ENST Bretagne/ICI, Nicolas.Jullien@enst-bretagne.fr,
MÉLANIE CLÉMENT-FONTAINE, Alcôve, Université de Montpellier I/ERCIM, melanie@amberlab.net.

Une «nouvelle économie du logiciel» semble apparaître, à la suite l'irruption des logiciels libres[1] et, dans une moindre mesure, de la relance du modèle des composants logiciels, avec ses incertitudes en terme de «modèle(s) économique(s)», avec aussi un questionnement sur le type de protection intellectuel le plus adapté à la production de logiciel. Ces questions se posent d'un point de vue général : quelle est l'évolution de l'industrie informatique, mais aussi d'un point de vue plus régional : comment l'Europe et la France peuvent profiter de cette évolution pour renforcer leurs positions dans les domaines où elles sont déjà fortes (services informatiques, systèmes embarqués, etc.) et les améliorer là où elles le sont moins (progiciels).

Cela revient à s'interroger sur la façon dont on produit du logiciel, dont on encadre et encourage cette production. Or, il n'existe, en économie du moins, que peu de travaux sur le sujet[2].

À cause des spécificités du bien «logiciel», cette industrie possède quatre caractéristiques originales (Richardson [1997]), qui expliquent la constitution de monopoles, mais aussi le dynamisme et le renouvellement continuel des entreprises productrices de logiciels. Conséquence directe du fait que c'est un bien «public[3]», le coût de développement d'un logiciel n'est pas affecté par l'ampleur de la population des utilisateurs et le coût d'extension marginal de cette population est nul ou réduit à un montant négligeable vis-à-vis du coût de développement. Le rythme de l'innovation est élevé, ce qui réduit la durée de vie des produits. Ces deux caractéristiques engendrant une concurrence forte, parfois agressive au niveau des prix, pour imposer sa solution et pouvoir jouir d'une rente de monopole. Les deux autres spécificités renforçant parfois ce type de concurrence, mais le limitant aussi : il existe des «effets de réseau», dans la mesure un logiciel «n'est d'aucune utilité en lui-même, mais seulement quand il est mis en oeuvre conjointement avec d'autres produits complémentaires au sein d'un système». Les entreprises qui possèdent un logiciel sont alors tentées de développer les logiciels complémentaires. Mais, en même temps, elles peuvent difficilement couvrir l'ensemble du spectre de la demande et de nouvelles entreprises se créent sans cesse pour répondre à de nouveaux besoins. La conséquence en est que les standards jouent un rôle très important car ils permettent de mettre en oeuvre conjointement des produits complémentaires. Là encore, la définition de standard favorise l'entreprise qui contrôle leur évolution, elle peut plus facilement anticiper sur leur évolution et garantir l'interopérabilité des programmes complémentaires. D'un autre côté, le fait qu'elle ne puisse pas répondre à l'ensemble des demandes l'oblige à rendre publiques leurs caractéristiques.

Avec le développement des progiciels, la constitution des monopoles s'est exacerbée, en même temps que se réalisait la diffusion de l'ordinateur, jusqu'à en faire aujourd'hui un outil « grand-public». Or, comme l'explique Horn [2000b], si cela s'est accompagné d'indéniables progrès en terme de productivité et de qualité de la production, ils sont insuffisants par rapport à l'augmentation de la quantité et à la diversité des logiciels à produire, aux exigences croissantes des utilisateurs, et à la complexité grandissante de ce qui est informatisé.

En réponse à ces insuffisances des développeurs, et d'abord Richard Stallman, ont proposé un système de production alternatif, que nous appellerons le «Libre», ou les logiciels produits sont mis à disposition de leurs utilisateurs, avec leurs codes sources et le droit de les modifier et de les rediffuser sans contrepartie. La diffusion d'Apache, de Sendmail ou, plus récemment, de Linux montrent le succès de cette initiative. Mais le Libre, qui empêche le financement de la production de logiciel par la vente de licences (puisque chacun peut en faire des copies) remet en cause cette organisation de production. Pourtant il ne semble pas s'opposer au marché, à la production et à la distribution marchande de logiciel, puisque de plus en plus d'entreprises, comme IBM, intègrent des logiciels libres dans leurs offres et participent à leur développement[4]. Ce

[1]Que nous défi nissons comme un logiciel distribué avec son code source et avec l'autorisation de le modifi er et de le redistribuer librement. Nous appellerons «Libre» le système de production des logiciels libres.

[2]Mowery [1996] (p. 312), l'un des rares auteurs ayant travaillé sur le sujet, écrit d'ailleurs que «l'industrie du logiciel a peu attiré l'attention de manière surprenante, étant donné sa taille, sa croissance rapide et son importance de plus en plus grande dans les industries de haute technologie»

[3]C'est-à-dire un bien qui n'est pas détruit par l'usage et qui est «non exclusif» (sa consommation par une personne n'empêche pas sa consommation par une autre, grâce aux copies, qui se font à coût quasi nul).

[4]SUN est, par exemple, le deuxième contributeur à des projets libres, en terme de code source produit, d'après Orbiten [2000]

serait un système qui réussirait à articuler une production non-marchande avec des activités marchandes complémentaires, proposant ainsi une nouvelle façon de développer le «bien» logiciel.

Or «c'est en définitive l'optimisation des activités économiques contraintes par les règles de droit qui conduit à la définition des biens échangés et donc à leurs différentes propriétés, destruction ou non par l'usage, exclusion ou non d'usage, obligation ou non d'usage. La définition des biens est endogène et non exogène» (Laffont [1991], p. 42). Et effectivement, comme nous le montrons dans le chapitre introductif, la place du logiciel dans l'activité informatique, composant de la machine, service, bien, et donc la façon de le définir par le droit, ont varié au cours du temps.

L'émergence du Libre demande que l'on s'interroge sur la possibilité que cette cohabitation, originale, d'une production non-marchande avec une distribution marchande, soit la base d'une nouvelle organisation de cette activité économique, fondées sur de nouvelles règles juridiques. Cette étude est aussi importante du point de vue de la théorie économique et juridique, les chercheurs apparaissant parfois en retrait sur ces réflexions, peut-être parce que leurs modèles habituels ne sont plus adaptés à un environnement nouveau, émergent et en pleine évolution. Enfin, si tant est qu'on puisse juger qu'il s'agit là de modèles économiques performants, et considérant la place de premier rang des entreprises européennes[5], il faut définir les outils institutionnels à mettre en place pour garantir que ces modèles soient soutenables et durablement créatifs.

C'est dans ce contexte qu'il faut entendre le projet «Nouveau Modèle(s) Économiques, Nouvelle Économie du Logiciel tel qu'il a été labellisé par le RNTL. C'est pourquoi, dès le premier workshop organisé en décembre 2000 avec la collaboration de l'INRIA nous avons mis en présence des praticiens et des chercheurs issus de différentes origines, expérience que nous avons renouvelée en mars 2002. La qualité des débats a été à la hauteur de nos attentes et a contribué à l'alimenter en travaux et en réflexions originales, destinés non seulement aux chercheurs mais aussi aux décideurs. Ce sont en grande partie ces travaux que nous présentons dans ce rapport.

Ses trois parties correspondent aux trois étapes de notre réflexion : décrire les modèles économiques, expliquer comment on peut construire une stratégie industrielle de valorisation par la mise en libre d'un logiciel, constater la nécessité de maîtriser le cadre juridique qui autorise ces stratégies et l'expliciter.

Parce qu'il s'agissait d'étudier les nouveaux modèles économiques du logiciel, la première préoccupation de notre groupe de travail a été d'étudier l'évolution des métiers informatiques qu'entraîne le Libre. Pour rapporter ces travaux, nous avons choisi de présenter, sous forme d'interview ou de monographie, des entreprises qui nous ont semblé représentatives des différents «métiers» de l'informatique et de la façon dont ces métiers s'approprient ces nouveaux «outils», en terme de technique, mais aussi de relation commerciale. Réalisées par Aymeric Poulain-Maubant (Jipo Interactive), ces études se sont intéressées à IBM, ACT Europe (éditeur du compilateur Ada 95 «Gnat»), MandrakeSoft (éditeur de la distribution GNU/Linux éponyme) et Makina Corpus (société de service).

Constatant ces différents modèles, on peut se demander pourquoi une entreprise productrice de logiciel va choisir l'un ou l'autre des ces modèles libres, et surtout pourquoi elle choisirait parmi ceux-ci plutôt que le modèle classique, de licence payante d'un logiciel fermé. Pourquoi, comment des producteurs de logiciels valorisent le fait de diffuser un logiciel sous une licence libre ? La deuxième partie du rapport s'intéresse à cette question, en proposant deux approches.
Considérant l'importance des sociétés de service dans la diffusion des logiciels libres et dans la construction d'un marché autour de ceux-ci, considérant aussi le fait qu'elles représentent la grande majorité des acteurs marchands du Libre, il nous a semblé important de commencer par proposer une analyse de ce secteur. C'est

[5]Comme l'a soulignée Jean-Michel Dalle lors du second workshop, nous sommes en avance quant à l'utilisation de ces logiciels : une illustration en est que Linux est la «marque» nouvelle économie la plus connue en France et en Allemagne, alors qu'elle n'est même pas dans les 20 premières aux USA

ce qu'a réalisée Marie Coris (Université Montesquieu, Bordeaux IV), dans son article intitulé «les sociétés de service en logiciels libres : l'émergence d'un système de production alternatif au sein de l'industrie du logiciel ?
Si ces entreprises se sont, le plus souvent, construites à partir d'initiatives de développeurs de logiciels libres, ce mode de production ne se développera que si des acteurs traditionnels l'adoptent et notamment si des entreprises, déjà existantes, font la démarche inverse qui consiste à intégrer la publication de logiciels sous licence libre. C'est ce qu'a étudié François Horn (Université de Lille III) au travers de deux exemples de mise en libre de logiciel, dans son article «Les stratégies de libération du code source d'un logiciel par une entreprise : opportunité et difficultés. Premiers enseignements de deux exemples récents dans le secteur de la CAO et de la simulation numérique». Il y explique, notamment, l'importance des contraintes organisationnelles, mais aussi la nécessité d'une réflexion ex-ante sur le positionnement marchand de l'offre et donc sur les caractéristiques de la licence à adopter.
Considérant ce deuxième point, Laure Muselli (Université de Paris XIII) propose une étude des différentes licences existantes en terme de positionnement marchand dans son article «Les licences : outils stratégiques pour les éditeurs de logiciels ?». Elle y explique comment les clauses des ces licences contraignent la relation avec les utilisateurs-développeurs, mais aussi la stratégie de valorisation financière, de manière statique (au moment où la licence est choisie), mais aussi de façon dynamique (comment on peut faire évoluer ces relations au fur et à mesure de la diffusion du logiciel).

Finalement, étudier les stratégies de valorisation, c'est s'intéresser aux rapports, souvent contractuels, que les producteurs de logiciel veulent construire avec leurs utilisateurs. Et ces rapports reposent sur la licence qui accompagne le logiciel et sont soumis à la législation en vigueur. C'est pourquoi nous consacrons la dernière partie de ce rapport aux questions juridiques qui encadrent la pratique du logiciel libre (utilisation et production) en France. L'étude des licences libres a été menée en vue de constituer un outil utile aux économistes et aux acteurs des domaines d'activités étudiés dans ce projet RNTL. Aussi, l'accent à été mis sur une présentation pratique et le rapport comprend le résultat des travaux menés sur le sujet selon trois angles différents.
Tout d'abord Yves Rougy (Alcôve) fournit, dans son article «étude des usages des licences libres» un panorama très complet de la pratique des licences : leur origine géographique, leurs auteurs (particuliers, établissements privés ou publics), le type de logiciels auxquels elles s'appliquent... Autant d'informations dont le traitement permet de mieux appréhender la pléiade de licences de logiciels libres et l'analyse juridique qui en est faite dans le deuxième chapitre de cette partie.
Cette analyse, est proposée par Mélanie Clément-Fontaine (Alcôve, Université de droit Montpellier, ERCIM), et s'intitule «étude juridique des licences libres». Malgré les nombreuses zones d'ombre dues notamment à l'absence de jurisprudence ou de texte officiel venant améliorer notre connaissance des licences libres, ce travail devrait permettre de mieux comprendre les notions juridiques rencontrées dans ce domaine, grâce aux différentes définitions proposées. Les questions communes à toutes les licences libres (la langue de contrat, les clauses de garanties...) sont expliquées et analysées. Enfin, les licences de logiciels libres les plus significatives font l'objet d'une description organisée sous forme de classification.
Par ailleurs, étant donné la place actuelle de la question de la brevetabilité des logiciels dans l'actualité juridique et de son implication économique, l'analyse de cette question a trouvé naturellement sa place dans le présent rapport, grâce au travail de Franck Macrez (Université de droit Montpellier, ERCIM), «la brevetabilité des logiciels».

Si l'ensemble de ces contributions montre le rôle aujourd'hui stratégique du Libre pour l'industrie, il souligne aussi l'importance du rôle du régulateur public. C'est lui qui oriente le système concurrentiel, en agissant sur le cadre légal de la protection intellectuelle, mais aussi en définissant des politiques industrielles visant à organiser ou à soutenir l'industrie française et européenne du logiciel. C'est pourquoi, dans la conclusion de ce rapport, Jean-Michel Dalle discutera des opportunités et des contraintes que l'évolution actuelle de l'économie du logiciel offrent à l'action politique[6].

[6]Cette conclusion sera présentée à la remise du rapport au RNTL à la fi n du mois de novembre.

Enfin, et en l'occurrence en premier, nous avons voulu proposer un chapitre introductif, qui, en replaçant le Libre dans la perspective historique de l'évolution de l'industrie informatique, en rappelant du régime de protection des logiciels, permet de mieux comprendre l'apparition et le succès de cette organisation de production, l'originalité juridique des licences libres, et les enjeux économico-juridiques de cette diffusion. Nous espérons que ce chapitre permettra à chacun, spécialiste ou non du sujet, d'avoir une vue générale de ce phénomène et, ainsi, de pouvoir apprécier les contributions plus pointues qui constituent ce rapport.

Le contexte économique et juridique de l’émergence d’une nouvelle économie du logiciel.

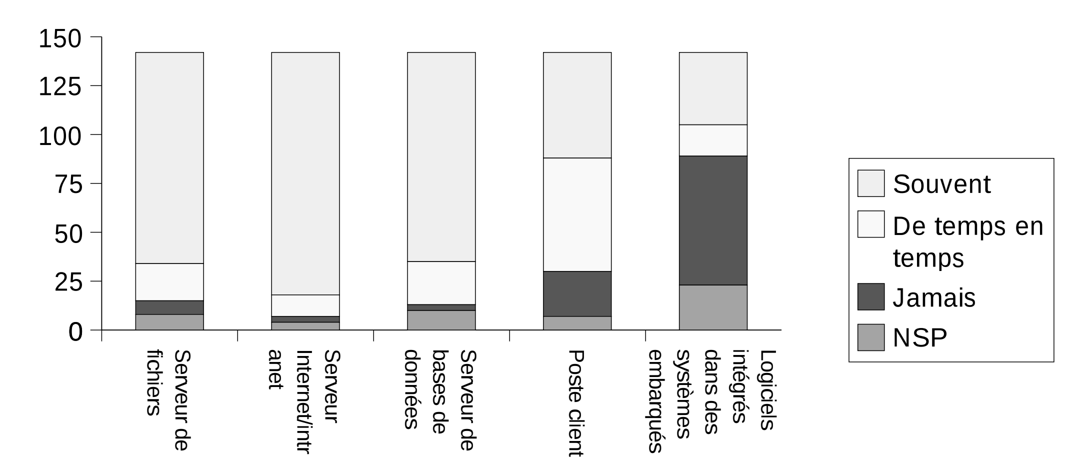
Nicolas Jullien, ENST Bretagne/ICI, Nicolas.Jullien@enst-bretagne.fr, pour la partie économique
Mélanie Clément-Fontaine, Alcôve, Université de Montpellier I/ERCIM, melanie@amberlab.net, pour la partie juridique.

Ce chapitre introductif a deux objectifs : replacer le système de production des logiciels libres (le «Libre») dans la perspective de l'évolution de l'industrie informatique afin d'expliquer son apparition et son succès actuel et, à partir de là, proposer une analyse synthétique de son évolution possible, analyse qui sera ensuite approfondie dans les différentes contributions de ce rapport. Nous consacrerons la première partie aux conditions d'émergence du phénomène des logiciels libres, ce qui nous amènera nous amènera à ce qui est au cœur de ce rapport : l'étude des stratégies poursuivies par les utilisateurs et les producteurs de logiciels libres, à laquelle nous consacrerons la deuxième partie. Enfin, nous envisagerons le contexte juridique dans lequel s'inscrit l'émergence des licences de logiciels libres.

2.1 Le logiciel libre, révélateur et moteur de l'évolution vers les services et la notion de qualité dans le logiciel.

Le succès actuel des logiciels libres semble reposer d'abord sur une adéquation historique à de nouveaux marchés et spécifiquement à la diffusion d'Internet[1]. Aujourd'hui encore, les offres commerciales construites à base de logiciels libres concernent plutôt des intranets/internets que des serveurs de fichiers, de bases de données ou des postes clients, même si cette diffusion est en cours, comme le montre le graphique ci-dessous[2].

Figure 2.1 **—** Types de solutions construites avec des logiciels libres.



Mais quand bien même les logiciels libres se diffuseraient, il serait hasardeux d'en conclure que l'organisation de production libre, coopérative, est adoptée : le précédent du système d'exploitation Unix qui a d'abord été développé de façon ouverte, coopérative, avant d'être privatisé et décliné en différentes versions «propriétaires» est là pour nous mettre en garde. Une analyse plus détaillée ne nous semble donc pas inutile pour expliquer pourquoi la diffusion du «Libre», présentant toutes les caractéristiques des précédents changements de structure industrielle en informatique, peut être envisagée comme la nouvelle base de la production industrielle de logiciels.

[1] Le site http://www.netaction.org/articles/freesoft.html rappel l'importance des logiciels libres dans le fonctionnement actuel d'Internet. Voir aussi Jullien [1999], 2001 sur le lien entre diffusion d'Internet des logiciels libres.

[2] Ce graphique, ainsi que les suivants, est tiré d'une enquête réalisée par Nicolas Jullien auprès des entreprises françaises proposant des produits (biens ou services) basés sur des logiciels libres (réalisée entre mai et début juillet 2002, 141 réponses). L'analyse de ces données est en cours, les résultats seront publiés sur le site http://www-eco.enst-bretagne.fr/Etudes_projets/RNTL/ au cours de l'automne.

2.1.1 L'évolution du système de production en informatique ; un phénomène rendu possible par les progrès technologiques, initié par les producteurs, mais choisi par les utilisateurs.

La structure industrielle de l'informatique[3] a subit, en deux étapes, des changements majeurs : d'abord, suivant l'arrivée de la série 360, ont commencé à apparaître des fournisseurs de matériel et de logiciel indépendants des constructeurs ; ensuite, avec l'arrivée du PC, la production du système d'exploitation fut, elle aussi, séparée de la production des machines. Cette évolution ne s'explique pas par des progrès conceptuels majeurs sur ce qui caractérise un ordinateur : aujourd'hui encore, ils sont fondés sur les principes posés par von Neumann pour la construction du premier ordinateur, mais essentiellement par les progrès sur les composants de base, essentiellement la miniaturisation de ces composants[4]. Et ces progrès ont eu deux conséquences dont la combinaison explique les évolutions de l'industrie. D'abord, cela a permis la baisse du prix des machines et donc leur démocratisation. En second, parce que la puissance des machines a augmenté, parce que les technologies logiciel ont, elles aussi, fait des progrès, on a pu complexifier le lien entre le logiciel permettant de répondre à un service d'usage et la machine sur laquelle il s'exécute.

Des désintégrations verticales rendues possibles par l'évolution des technologies «logiciel».

Les compilateurs ont permis, à partir des années 50, de séparer la façon dont les instructions sont codées pour la machine de la façon dont elles sont exprimées par le programmeur. à partir de là, on a pu réaliser un programme pour une machine et le réutiliser sur une autre machine, dès lors qu'il existait un compilateur adapté à cette nouvelle machine. La création de langages de programmation standard (LISP, Fortran) initia l'idée qu'on pouvait construire des logiciels indépendamment de la machine, ce qui conduit à les concevoir comme des produits, ce qui n'était pas le cas aux débuts de l'informatique. Mais c'est avec l'invention du concept de système d'exploitation, qu'on peut séparer réellement la production de la machine de celle du logiciel d'utilisation. En effet, on peut alors faire évoluer son logiciel, augmenter ses fonctions, en étant sûr de trouver une machine plus puissante, qui le fasse fonctionner, car le système d'exploitation devient un standard commun à toute une gamme d'ordinateurs. C'est la première possibilité de désintégration verticale de l'industrie, entre la machine d'une part et les logiciels d'utilisation d'autre part.

La deuxième possibilité est ouverte, au début des années 80, par l'arrivée du micro-processeur, qui regroupe l'ensemble des éléments qui font les calculs dans un ordinateur, autorisant une séparation entre la production de ce composant et celle du système d'exploitation car les instructions peut envoyer le système d'exploitation à ce centre de calcul sont, à leur tour, standardisées. On a, à nouveau, séparation possible entre production de la partie matérielle de la machine et du logiciel (le système d'exploitation) qui permet de faire fonctionner cet assemblage, avec la possibilité d'avoir plusieurs producteurs de matériel pour un même logiciel d'exploitation. La désintégration industrielle a effectivement suivi cette désintégration technologique parce que des producteurs (en fait, IBM dans les deux cas) l'ont provoquée en proposant de nouveaux modèles de production, accompagnant de nouveaux produits (la série 360 et le PC)[5]. Mais le positionnement stratégique des producteurs n'explique, seul, l'évolution que l'on constate sur la façon dont est perçu le logiciel, passant d'un composant de la machine que l'on s'échangeait volontiers à un produit industriel protégé par

[3]Notre compréhension de l'histoire de l'informatique doit beaucoup aux ouvrages de Breton [1990], Genthon [1995] et de Dréan [1996]. L'analyse de l'organisation de l'industrie informatique s'appuiera aussi sur le travail de Gérard-Varet et Zimmermann [1985], initié dans Delapierre et al. [1980] et développé par Zimmermann [1989], 1995a et Delapierre et Zimmermann [1994]. Enfi n, notre analyse de l'économie et de l'industrie du logiciel doit beaucoup à Horn [2000b], qui nous apparaît comme l'ouvrage de référence dans le domaine. Nous encourageons le lecteur désireux d'approfondir ces sujets à se reporter à ces travaux.

[4]Passant des lampes à vide, puis des transistors, à des technologies propres à l'informatique, le circuit imprimé puis, grâce à la miniaturisation, les circuits intégrés.

[5]Pour se convaincre de l'importance de ces initiatives industrielles, il suffi t de se rappeler, avec Genthon [1995], que la séparation entre producteurs des machines personnelles et de systèmes d'exploitation n'est pas une norme universelle : elle n'existe pas au Japon et Apple ou SUN dessinent toujours leurs micro-processeurs et produisent leur système d'exploitation.

des licences d'utilisation très strictes. Après tout, ce sont ceux qui produisaient les logiciels coopérativement qui ont inventé le système d'exploitation ou le micro-processeur. Et, a priori, le fait de séparer la production des logiciels des machines rendait plus facile leur réutilisation. Cela aurait dû favoriser l'échange et la coopération, système dominant aux débuts de l'informatique. C'est, pour nous, l'évolution de la demande (due à l'évolution des utilisateurs), qui explique le succès de ces nouvelles offres industrielles.

Des modèles d'organisations industrielles sélectionnés par la demande.

Le fait que l'objet informatique soit un assemblage complexe est source d'inégalité entre les utilisateurs : certains sont capables de comprendre comment il est architecturé quand d'autres n'ont comme connaissance de son fonctionnement que la façon de l'utiliser. Or, à cause des phénomènes de standardisation, le choix des utilisateurs a une influence sur l'offre disponible, mais aussi sur la demande des autres utilisateurs. Par exemple, plus il y a de personnes qui choisissent un système d'exploitation, plus l'offre de logiciels pour ce système se développe et plus il devient intéressant de le choisir. L'ordre dans lequel les adopteurs choisissent les technologies est important, a une influence sur l'offre finalement disponible et, partant, sur les modèles industriels qui les supportent. Bien sûr, cela a aussi un impact fort sur les stratégies des producteurs, qui cherchent à imposer leur offre comme le standard, comme nous le rappelions dans l'introduction. D'ailleurs, nous avons souligné que les évolutions des organisations industrielles sont d'abord dues à la construction par ces derniers de nouvelles stratégies et de nouveaux systèmes de production. Mais c'est bien parce que les utilisateurs ont évolué, faisant évoluer leurs besoins, que ces nouvelles organisations s'imposent.

Ainsi, aux débuts de l'informatique, les clients étaient souvent des militaires ou/et des centres de recherche, qui passaient des contrats proches de contrats de recherche. Les utilisateurs avaient tous un très haut niveau d'expertise technique. L'évaluation des offres se faisait au niveau du matériel, des composants technologiques ; la relation client-fournisseur était une relation de service, de co-définition des besoins et de co-construction de la réponse, dans une culture du prototype. Mais la diffusion de ces machines dans les entreprises a fait évoluer les contraintes, intégrant des critères comme l'amortissement des investissements logiciels sur une plus grande période, donc aussi le caractère réutilisable, évolutif de ces logiciels. La possibilité d'accumuler des connaissances, de développer sans cesse de nouvelles fonctionnalités rendait aussi ces logiciels de plus en plus importants pour les entreprises, mais aussi de plus en plus complexes. Ils requerraient de plus en plus de connaissances de la part des développeurs, en même temps que baissait, chez ces utilisateurs-entreprises la maîtrise des connaissances techniques sur les composants utilisés dans les machines. Parallèlement le poids des dépenses publiques diminuait, donc aussi le pouvoir de prescripteur que les utilisateurs-chercheurs avaient sur la filière. On comprend alors l'intérêt du modèle industriel reposant sur une gamme de machine, utilisant le même système d'exploitation, permettant d'ignorer l'architecture physique sous-jacente et sur la création d'entreprises de service spécialisées dans le développement de logiciels d'application, spécifiques aux entreprises. Et c'est bien ce modèle qui s'est imposé au cours des années soixante-dix. On est passé d'un **logiciel-composant de la machine à un logiciel-produit à façon pour l'entreprise**.

Enfin, la troisième période, celle de la micro-informatique, a débuté au milieu des années 1970, avec l'idée de construire des machines individuelles (permettant aux utilisateurs de s'affranchir du pouvoir, souvent tout puissant des «directions informatiques»). Les mécanismes du succès du PC d'IBM sont similaires à ceux qui ont expliqué le succès du modèle instauré par la série 360 d'IBM. Chacun des deux éléments clefs de la machine, le microprocesseur et surtout le système d'exploitation, n'a rapidement été produit que par une entreprise, en situation de quasi-monopolee, condition nécessaire pour garantir la pérennité des investissements et comparer facilement les offres de machines. Car la spécificité de cette diffusion résidait dans le fait que plusieurs entreprises pouvaient produire les mêmes machines, ce qui a créé une concurrence forte sur le matériel et a permis une évolution plus rapide du rapport prix/performances que dans l'ancien système. Ensuite, ces machines se sont diffusées pour répondre à de nouveaux usages, de traitement de l'information personnelle (tableurs et traitements de texte). Enfin, ce sont les entreprises qui ont d'abord acheté ces machines, avant que

leur utilisation au bureau et la baisse des prix ne permettent leur diffusion chez les particuliers. Le caractère peu spécifique des usages de ces machines, le fait que la concurrence soit essentiellement une concurrence en prix expliquent le développement des progiciels (Mowery [1996]). Cela, et son corollaire, la tendance au monopole, ont été accentué au cours du temps, au fur et à mesure que les utilisateurs novices[6] devenaient prescripteurs, parce que leurs connaissances, en plus de la marque, se réduisaient aux fonctionnalités et à l'ergonomie d'un produit, qui devenait alors coûteux à remplacer. Le logiciel est passé de **produit-spécifique à produit-standard**. La conséquence est que l'offre s'est rapidement standardisée autour de quelques produits par fonction à cause des caractéristiques particulières du logiciel : il est sans coût de reproduction, donc plus son utilisation est partagée par un grand nombre de personne, plus la part du coût de développement supportée par chaque utilisateur est faible.

Des mécanismes d'évolution stables, mais peu favorables à la production coopérative.

On peut donc dire, en guise de conclusion partielle, que, grâce au progrès technologique, des utilisateurs de plus en plus nombreux ont eu accès à l'objet ordinateur ; que ces utilisateurs, étant de moins en moins compétents en informatique ont eu besoins de tiers pour développer les logiciels qui répondaient à leurs besoins en terme de technologies d'utilisation ; que l'arrivée de nouveaux utilisateurs a sélectionné les modèles, les positionnements industriels qui rendaient l'achat de l'objet informatique le plus simple possible pour eux. La simplicité a été synonyme de développement et de vente de logiciels standards par des producteurs principalement parce que ces utilisateurs n'avaient pas les compétences pour les développer et les évaluer. Une conséquence de cette marchandisation progressive a été la marginalisation de la production coopérative. Après cette analyse, le Libre, organisation coopérative de production de logiciels sur des bases essentiellement techniques, semble d'autant plus improbable, voire anachronique. Pourtant, celle-ci a aussi montré qu'avec l'évolution de la demande de nouvelles organisations de production pouvaient apparaître. D'autre part, la production coopérative a continué à exister et à apporter de nouvelles technologies et de nouveaux usages aux producteurs et aux utilisateurs de l'outil informatique (Unix, Internet et ses services, pour ne citer qu'eux). C'est, une fois de plus, parce que les technologies et les demandes ont à nouveau évolué que cette organisation propose, aujourd'hui, une solution cohérente et intéressante pour le plus grand nombre.

2.1.2 Le Libre, nouvelle organisation de production d'une nouvelle informatique ?

Progrès technologiques et évolution de la demande.

Au cours des années 1990, la principale évolution technique en informatique a, bien sûr été, avec l'arrivée d'Internet, la généralisation de la mise en réseau des ordinateurs, à l'intérieur et à l'extérieur des organisations. Mais la miniaturisation a, une fois de plus, permis l'apparition d'une nouvelle gamme de produits, «nomades» («organisers» (Psion et Palm), téléphones mobiles). Cela s'inscrit dans la continuité de l'évolution des produits informatiques : on est passé de la machine unique, dédiée à une tâche connue par avance et dévolue à l'ensemble d'une organisation, à des machines multiples, utilisées pour réaliser des tâches variées et variables dans le temps, et intégrées dans une organisation et communicantes.Mise en réseau, échange entre systèmes hétérogènes, le problème de la communication entre ces machines devient crucial, d'autant plus qu'elles se sont multipliées. Pour le résoudre, il faut mettre au point des standards au niveau du réseau d'échange, mais aussi au niveau des données échangées, afin qu'elles soient transmises et qu'elles soient déchiffrables par tous ces systèmes. Ainsi, le progrès d'Internet n'est pas d'avoir proposé un «protocole» pour permettre la transmission simple des données, il en existait déjà, mais d'en avoir proposé un suffisamment simple, suffisamment souple, pour s'être imposé comme un standard d'échange. Les technologies logiciels ont, elles aussi, évoluées (Horn [2000b], pp. 126-128) : l'arrivée des langages de programmation objet (Smaltalk, C++, Java et

[6] «Naïfs», pour reprendre la terminologie de Delapierre et Zimmermann [1994].

peut-être bientôt C#) a facilité la réutilisation de composants logiciels déjà développés ; cela a débouché sur le concept du «logiciel modulaire» dont l'idée est de développer un ensemble de petits logiciels (des modules ou des composants logiciels), qui rendraient chacun un service précis et qui seraient associables, utilisables sur n'importe quelle machine car leurs interfaces de communication seraient standards. Cette réutilisation, dont on parle depuis déjà longtemps, est rendue aujourd'hui plus aisée par les derniers progrès du génie logiciel (Beugnard [2001]), qui permet de mieux spécifier le fonctionnement et les protocoles d'échanges entre composants. Ce qui caractérise l'évolution technologique dans le logiciel est donc l'interdépendance croissante entre les logiciels, en même temps que la spécialisation de plus en plus fine des composants logiciels réutilisés (Zimmermann [1998]). Ce système ne peut fonctionner que si les composants sont effectivement réutilisables, c'est-à-dire si les producteurs s'accordent autour d'un mécanisme qui permette de standardiser les interfaces et de garantir, dans le temps, la stabilité des standards. Cette évolution n'est pas non plus sans conséquence sur les caractéristiques de la demande. La mise en réseau des utilisateurs a accentué le besoin que des logiciels, souvent produits par des entreprises différentes, pour des utilisateurs différents, puissent échanger des données. Il faut aussi qu'elles s'assurent de sa disponibilité dans le temps, malgré les changements de version. Il devient donc nécessaire de standardiser les formats d'échange des fichiers. Il est alors logique que les entreprises recherchent des solutions plus ouvertes, qui leur garantissent de mieux disposer de ce contrôle. Ensuite s'exprime le besoin d'une nécessaire adaptabilité des logiciels. En effet, l'hétérogénéité de la demande interne aux entreprises s'est aussi accrue avec la mise en réseau des différents systèmes et avec la nécessité que des utilisateurs de l'entreprise partagent les mêmes outils. Il faut alors adapter les logiciels (et particulièrement les progiciels) aux besoins et aux connaissances de chacun. Dernier point, la composition des groupes d'utilisateurs a aussi évolué, essentiellement pour des raisons de coût : des utilisateurs ayant des compétences techniques fortes utilisent de plus en plus les progiciels, contribuant à élever le niveau moyen d'exigence vis-à-vis de ces produits. Les utilisateurs naïfs ont aussi pu se rendre compte, à l'usage, de l'inadéquation de certains logiciels avec leurs besoins : l'existence d'un nombre important de fonctionnalités inutiles rend, par exemple, le logiciel très gourmand en capacité de calcul, fait qu'il ne fonctionne pas ou mal sur les machines un peu anciennes et complexifie beaucoup son utilisation. La conséquence en est, qu'une fois de plus, les relations marchandes sont en train d'évoluer : les relations sont de plus en plus **des relations de service, basées sur l'adaptation de logiciels composants-standards**[7], à tel point que Horn [2000b] défend l'idée que nous serions entrés dans une nouvelle phase de la production, le «sur-mesure de masse». Mais ces relations de service souffrent d'une relativement inefficacité : lorsqu'on s'intéresse aux enquêtes de satisfaction sur les produits informatiques[8], on constate une satisfaction vis-à-vis de l'outil informatique, mais une insuffisance dans le service après-vente, surtout dans le logiciel. La tendance de fond est que le client «recherche un meilleur support avant et après-vente, une aide pour résoudre ses difficultés et satisfaire ses besoins». Or, le Libre apparaît comme une réponse, développée par certains utilisateurs, aux insuffisances des progiciels en terme de qualité et d'interopérabilité. Il semble aussi améliorer les relations de service.

Le Libre, un système construit par les utilisateurs développeurs avec l'objectif d'améliorer la qualité des logiciels produits et assurer le respect des standards.

Plus que des produits de la recherche publique, les logiciels libres sont d'abord des outils développés par des utilisateurs-experts, pour leurs besoins propres. La faible qualité des progiciels fermé et surtout la diffi-

[7]La part des services dans le chiffre d'affaires d'IBM est passée de 32 à près de 41 % en quatre ans (1997-2001, voir http://www.ibm.com/annualreport/2001/home/index.html), c'est le seul poste qui progresse signifi cativement. HP a tenté dernièrement de racheter la partie conseil de PriceWaterhouseCoopers, fi nalement acquise par IBM. Compaq, depuis rachetée par HP, avait annoncé en juin 2001 qu'elle se donnait 18 mois pour devenir une «service company», sans parler de Microsoft, qui avec sa nouvelle offre de licence et .Net essaie d'entrer sur ce marché.

[8]Nous nous référons ici à l'enquête de satisfaction réalisée chaque année par l'hebdomadaire 01 Informatique auprès des grands comptes français (enquêtes de 1998, numéro 1521, de 1999, numéro 1566 et de 2000, numéro 1612). Mais d'autres enquêtes existent, parfois plus sévères encore, comme celles de De Bandt [1995], ou de Dréan [1996] (p. 276 et suivantes).

culté pour les faire évoluer est une des raisons fondamentales de l'initiative de Richard Stallman[9]. Ceux-ci sont à l'origine, améliorent et contrôlent de nombreux logiciels (dont Linux, Apache ou Samba). Et on se doit de constater que, concernant ces logiciels phares, cette organisation[10] obtient des résultats remarquables en terme de qualité et de rapidité d'amélioration[11]. C'est sans doute d'abord dû à la libre disponibilité des sources, permettant aux utilisateurs d'effectivement tester les logiciels, étudier leur code et le corriger s'ils trouvent des erreurs. Or, plus il y a de contributeurs, plus la chance qu'un des contributeurs trouve une des erreurs augmente. Augmente aussi la chance qu'il y en ait un qui sache la corriger. Mais le Libre, ce sont aussi des outils (les langages) et des règles de programmation, qui rendent possible cette lecture[12]. Et tout ceci contribue à garantir des seuils minima de robustesse du logiciel. Le travail coopératif, le fait que les logiciels développés soient souvent un assemblage de petits projets évoluant en parallèle imposent aussi que les interfaces de communication soient publiques, normalisées. L'ouverture du code facilitant encore la vérification de cette compatibilité et au besoin, la modification des logiciels. Il est aussi remarquable de noter à ce propos que, pour éviter la reproduction de la divergence des versions d'Unix, les entreprises informatiques ont mis en place des organismes chargés de garantir la compatibilité entre les différentes versions et distributions de Linux et de publier des recommandations techniques sur la façon de programmer les applications pour qu'elles fonctionnent avec ce système, dans l'esprit de la norme POSIX[13]. Ceci a fait dire à Zimmermann [1999], à propos de la façon dont on peut assurer l'interopérabilité des logiciels, que, «si la solution des systèmes ouverts a été mise à mal par un récent retour aux cloisonnements des systèmes propriétaires, celle des logiciels libres pourrait contribuer à un dépassement des limites et contradictions des systèmes de propriété intellectuelle et par-là même impulser une profonde reconfiguration de l'industrie du logiciel». On a là un système performant du point de vue de la production des logiciels. L'est-il aussi du point de vu de l'adéquation des logiciels produits avec les besoins des utilisateurs ? Car pour qu'il s'agisse vraiment d'une organisation de production, il faut aussi qu'il soit adapté aux besoins, aux nécessités des utilisateurs (et notamment des entreprises, historiquement le maillon clef de l'évolution du modèle productif), qui n'ont plus toujours les compétences en interne pour évaluer, installer, suivre l'évolution de ces logiciels et les adapter à leurs besoins propres.

Le Libre, une organisation industrielle de production de services.

Ce n'est pas parce qu'un logiciel est librement disponible qu'il est accessible à chacun. Il faut savoir définir ses besoins, trouver un logiciel qui y réponde, l'installer, parfois l'adapter en développant des modules complémentaires. Une fois installé, il faut suivre son évolution. Tout cela nécessite de posséder des spécialistes de ces logiciels, ressource rare et dont les utilisateurs, notamment les entreprises ont rarement l'utilité à temps plein. Pour les servir, il est nécessaire que des compagnies se créent, et que cette activité soit rentable. Mais

[9]Qui a «inventé» le concept de logiciel libre, avec la création de la licence GNU/GPL, et créé la Free Software Fundation, organisation chargée de les produire ; voir http://www.fsf.org/gnu/thegnuproject.html.

[10]Sur la façon dont fonctionne l'organisation de production de logiciel libre, à côté de Raymond [1998a], 1998b, 1999, on pourra consulter Lakhani et von Hippel [2000] et Jullien [2001].

[11]La «communauté» des utilisateurs de Linux a reçu deux ans de suite le prix de la meilleure assistance technique (voir http://www.infoworld.com/cgi-bin/displayArchive.pl?/98/05/poy6a.dat.htm). Au sujet des tests de performance, on pourra consulter, pour les serveurs Web, http://www.syscontrol.ch/e/news/Serversoftware.html et pour les systèmes d'exploitation, http://gnet.dhs.org/stories/bloor.php3/. Les résultats de nombreux tests comparatifs sont disponibles sur les sites http://www.spec.org et http://www.kegel.com/nt-linux-benchmarks.html (ce dernier s'intéressant d'abord aux comparaisons NT/Linux). Enfi n, Wheeler [2001] a recensé nombre de données (test, parts de marché, etc.) sur les logiciels libres.

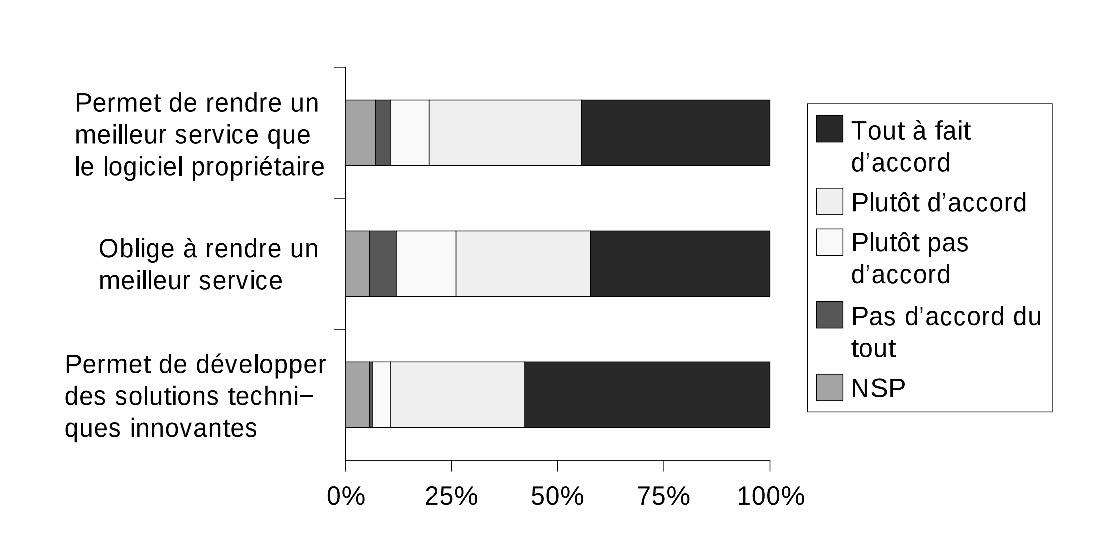
[12]On peut se souvenir que les premiers logiciels libres créés et adoptés ont été des outils de développement (et notamment des compilateurs, des interpréteurs et des outils d'aide à la programmation). Ce sont les logiciels qui permettaient de normaliser les langages informatiques, qui sont la base commune minimale nécessaire pour dialoguer. Voir http://www.gnu.org/prep/standards.html pour les recommandations techniques à suivre pour programmer des logiciels GNU.

[13]C'est le Free Standard Group (http ://www.freestandards.org/), qui regroupe le Linux Standard Base (LSB, www.linuxbase.org) et la Linux Internationalization Initiative (LI18NUX, www.li18nux.org). Parmi <FONT SIZE=2>les membres de ces comités, on trouve, entre autre, Corel, RedHat, Mandrake, SuSe, VA Linux Systems, Turbo Linux, mais aussi IBM, SUN, SAP, Caldera, SGI, ou l'Open Group (chargé de la standardisation sous Unix).

il faut d'abord que ces offres soient plus intéressantes que les solutions déjà existantes, le plus souvent basées sur des logiciels fermés. Enfin, il est nécessaire, pour ne pas casser la dynamique d'innovation qui existe dans l'industrie du logiciel, que les producteurs marchands aient des incitations à contribuer aux projets existants et à en initier de nouveaux. Bien sûr, l'absence de coût de licence donne un avantage concurrentiel non négligeable à la solution libre. Mais elle ne justifie pas, à elle seule l'adoption : il faut que la solution s'avère moins coûteuse à long terme, à qualité égale ; c'est cet indicateur qu'utilisent les fabricants de solutions propriétaires pour défendre leur offre[14].

Nous avons déjà dit que les logiciels libres les plus matures étaient de très bonne qualité. Cela facilite les relations entre producteurs de technologies d'utilisation et utilisateurs de ces mêmes technologies : les producteurs peuvent plus facilement garantir (contractuellement) la fiabilité des logiciels (libres) qu'ils utilisent, parce qu'ils sont capables d'évaluer cette qualité au travers des normes mises en place dans le développement, parce qu'ils disposent d'un réseau d'assistance et parce qu'il sont capables d'intervenir eux-mêmes sur les logiciels. D'autre part, le fait que l'on dispose des sources des logiciels, que l'évolution des logiciels ne soit pas contrôlée par une entreprise peut rassurer l'adopteur sur le fait que la solution respecte et continuera à respecter les standards, donc restera inter-opérable avec le reste des logiciels qu'il utilise. L'utilisation par des entreprises de services de logiciels libres peut être vue comme la création d'une norme professionnelle, puisqu'il s'agit, collectivement, de se coordonner pour choisir des composants, des briques logiciels fiables et surtout «normalisés[15]», ce qui fait actuellement défaut à l'industrie informatique (Dréan [1996]).

Figure 2.2 **—** Impact du Libre sur la relation de service.



La mise en commun des briques logiciels déplace aussi la concurrence des entreprises de service sur la relation de long-terme, la maintenance des logiciels. Il leur est aussi plus difficile d'invoquer une erreur du logiciel pour expliquer un mauvais fonctionnement du logiciel qu'elles ont installé et paramétré. C'est un élément qui peut inciter les entreprises à améliorer le service rendu au client, et qui nous fait dire que, à ce niveau là, les solutions libres peuvent être compétitives. Reste qu'il faut aussi que les entreprises puissent tirer un profit de leur activité, donc qu'elles contrôlent une partie de la valeur ajoutée. C'est sans doute le point le plus délicat à défendre, aujourd'hui : il y a peu d'exemple d'entreprises rentables et beaucoup qui n'ont pas encore atteint l'équilibre[16].

[14] C'est ce qui est appelé le «TCO», pour «Total Cost of Ownership». Microsoft, aujourd'hui, défend l'idée que si ses logiciels sont plus chers que les logiciels libres, ils ont un TCO inférieur, parce qu'il est plus facile de trouver des entreprises pour les installer, parce que leur évolution est garantie, contrôlée par une entreprise, etc.

[15] Au sens où elles respectent des formats publics et dont l'évolution est décidée collectivement.

[16] On peut citer, dans le premier groupe, ACT Europe, Easter Eggs. Mandrakesoft annonce l'équilibre pour le dernier trimestre 2002.

Nous retenons, cependant, les points suivants :

– au niveau des coûts de production, grâce à la construction en modules, le coût de développement d'un logiciel est plus réparti dans le temps, se rapprochant plus d'une structure de production de service où l'on ne développerait la fonctionnalité manquante qu'au moment où elle s'avérerait nécessaire. Les contributions des entreprises de services ne concernent pas la production complète d'un logiciel mais la production de ces composants pour des clients qui préfèrent que ces logiciels soient libres pour ne pas dépendre de leur fournisseur. Surtout, un composant développé pour un client peut être réutilisé pour répondre au besoin d'un autre et un «trou de sécurité» détecté chez un client peut être corrigé chez l'ensemble des clients de l'entreprise. Elles accaparent ainsi une partie des économies d'échelle engendrées par l'utilisation collective d'un logiciel, assurant, en échange, la diffusion des innovations et des corrections[17]. On peut dire des entreprises de services qui basent leurs offres sur des logiciels libres qu'elles proposent gratuitement les connaissances «codifiées» que sont ces logiciels pour vendre les connaissances «tacites» qu'elles possèdent : la connaissance du fonctionnement intime des logiciels, la capacité de leurs développeurs à produire des contributions qui fonctionnent, à faire accepter ces contributions par le noyau qui contrôle l'évolution des logiciels, etc. Ces entreprises sont les mieux placées pour accaparer les bénéfices d'apprentissage générés par le développement, l'amélioration de logiciels.
– à cause de ces effets d'apprentissage, à cause de la difficulté à diffuser les connaissances tacites qu'il faut maîtriser pour suivre et influer l'évolution d'un logiciel libre, ce rôle sera forcément réduit à quelques entreprises. Elles regrouperont en leur sein les spécialistes des logiciels et les mettront à disposition des entreprises clientes. Elles auront construit des marques fortes, reconnues par les utilisateurs-développeurs de logiciels et connues des autres clients. Cela permettra de diminuer la pression concurrentielle, assurant ainsi leur marge.

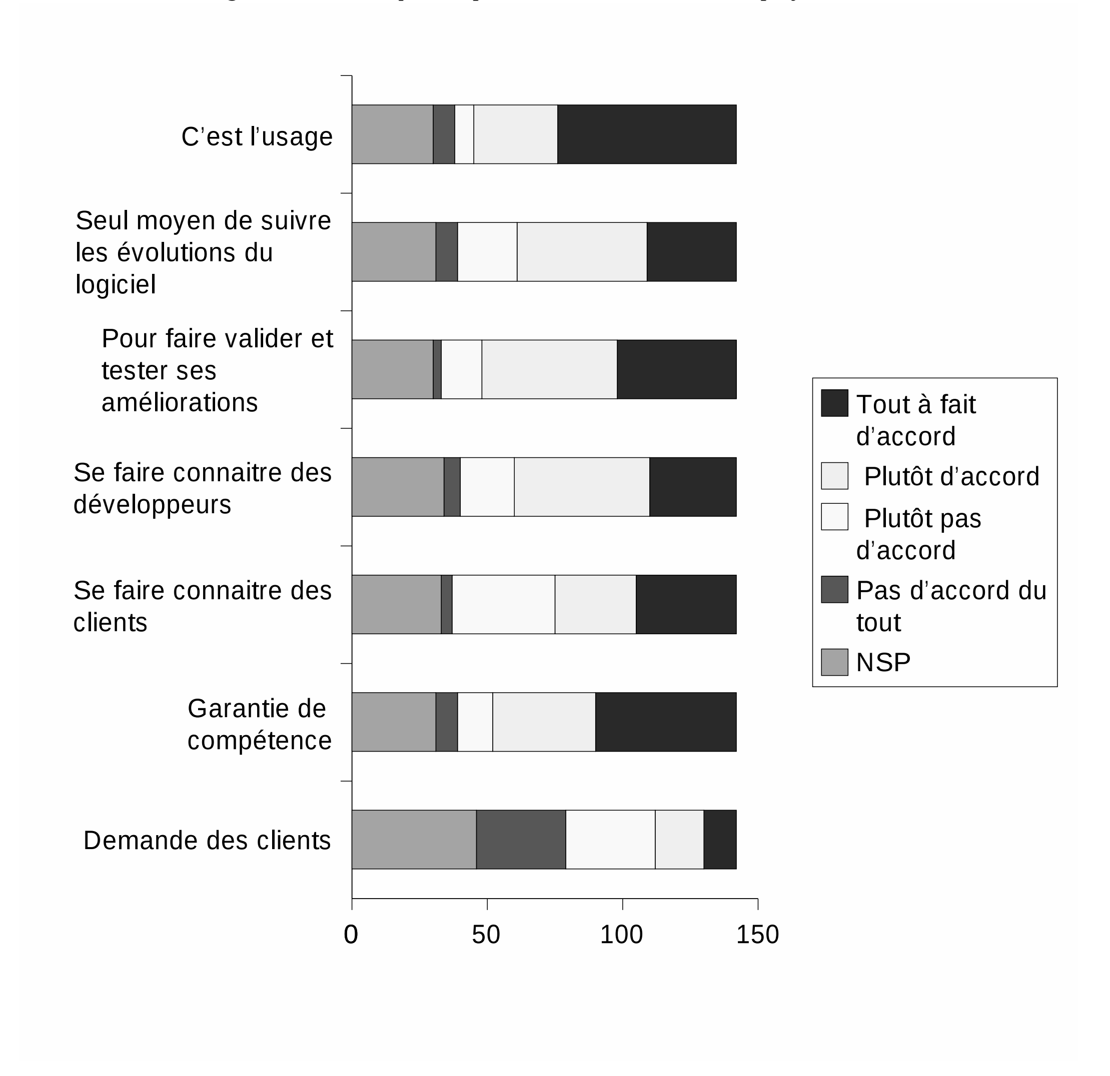
Ce type de concurrence incite aussi ces producteurs à contribuer au développement des logiciels qu'ils utilisent. C'est d'abord un moyen de se faire connaître, et de pouvoir signaler ses compétences en tant que développeur à ses clients. Et, parce que les besoins des clients sont différents, il est important pour ces entreprises de maîtriser un vaste porte-feuilles de logiciels, comme de contribuer au développement des logiciels standards utilisés dans la plupart des offres, pour pouvoir toujours présenter aux clients des réalisations en rapport avec leurs problèmes. Enfin, ne serait-ce que suivre les évolutions de ces logiciels est plus facile si l'on innove soit-même, cela permet de mieux comprendre les innovations des autres (Cohen et Levinthal [1989].)

Cela explique que ces entreprises consacrent une partie de leurs investissements à des développements de logiciels libres, sans que ces développements soient directement utiles (i.e. affectables à un contrat). C'est une activité de recherche et c'est une attitude qui les différencie des entreprises de service «classiques» qui consacrent souvent moins de 1 % de leur chiffre d'affaires à la recherche (voir Genthon [2001]).

Dans un marché fondé sur la valorisation de l'expertise technique, cette activité de contribution renforce l'image de l'entreprise sur sa capacité d'expertise et de réactivité, les deux qualités qui fondent une spécificité et qui permettent de singulariser l'offre, d'augmenter la notoriété (via la marque), et donc la marge. A contrario, cela renforce encore les tendances à la concentration, car il faut amortir ces coûts de recherche, donc augmenter le nombre de projets et de clients. On aurait ainsi un système qui, par son ouverture, facilite la relation de service, permet la production de logiciels tout en garantissant des revenus à long terme pour les fournisseurs. évidemment, de même que les évolutions précédentes n'ont pas éliminé les organisations anciennes, mais réduit leur champ d'application, de même le Libre n'est pas la solution universelle. Suivant les types de logiciel, suivant les types d'utilisation, l'organisation actuelle peut rester la plus efficace. On peut même imaginer des solutions hybrides, comme celles que propose SUN ou Troll Tech avec des systèmes de licences semi-libres ou des doubles licences (voir l'article de Laure Muselli). Nous finirons donc cette partie en précisant les critères favorables à la publication d'un logiciel sous une licence libre.

[17]On retrouve là un des rôles, classique, des éditeurs de logiciels, qui, traditionnellement financent cette activité en produisant et vendant de nouvelles versions du logiciel.

Figure 2.3 **—** Pourquoi les producteurs contribuent à des projets libres.

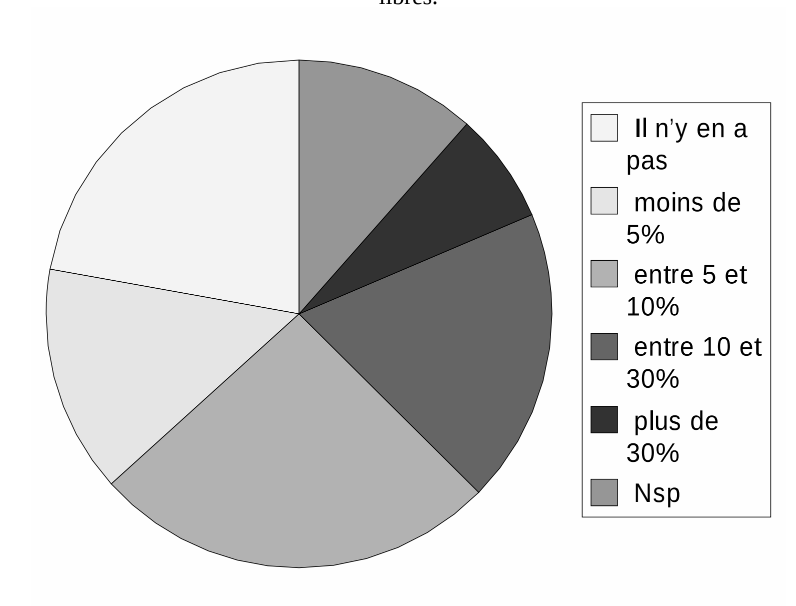


Quels marchés pour l'industrie du Libre ?

Les logiciels de réseau sont les archétypes des logiciels pour lesquels une publication libre est efficace : ce sont des logiciels utilisés par les informaticiens, ils demandent une standardisation forte et ils évoluent rapidement. L'existence d'utilisateurs avancés permet d'espérer des contributions, qui diminuent les coûts de développement, de correction des bogues ou de création des nouvelles fonctionnalités. Si ce groupe existe potentiellement, encore faut-il qu'il franchisse le pas de la participation active. Le fait que le logiciel soit en réseau, soit un standard, plaide pour sa mise en libre, afin d'être assuré qu'il respecte des normes. La dynamique d'innovation aussi : du côté des utilisateurs-producteurs, plus le logiciel évolue rapidement, plus on a intérêt à divulguer ses améliorations car cette dynamique risque de le rendre obsolète (si les mêmes fonctionnalités sont développées par d'autres) ou incompatible avec les nouvelles versions du logiciel, ce qui augmente le coût de maintenance. A contrario, plus le logiciel développé est important dans la stratégie d'une entreprise, plus elle aura tendance à le garder secret. Ces utilisateurs-producteurs vont principalement contribuer au développement des logiciels évoluant vite, assez peu spécifiques à une profession (les contributions n'apportent pas d'informations sur les choix technologiques ou industriels de celui qui les fait), ou très hétérogènes (les contributions seront difficilement utilisables ailleurs que dans l'organisation qui les a proposées).

Du côté des producteurs, ce sont aussi ces logiciels qui apportent le plus d'opportunités pour développer

Figure 2.4 — Temps de travail laissé aux employés pour contribuer personnellement à des projets libres.



des services (suivit du logiciel, création de nouvelles fonctionnalités et intégration de celles-ci dans la version officielle, etc.) Les utilisateurs sont aussi capables d'évaluer la qualité technique des développements proposés par les producteurs et de proposer leurs propres contributions. Notons que c'est surtout vrai lorsqu'il existe une offre propriétaire, dominante, la diffusion en Libre permettant plus facilement aux utilisateurs d'essayer cette alternative. Partant de cette analyse, on peut essayer de décrypter le positionnement des producteurs de logiciels libres, entreprises commerciales, mais aussi utilisateurs ou centres de recherche.

2.2 Produire et valoriser des logiciels libres.

Nous allons nous intéresser ici aux motivations des organisations productrices de logiciels libres, les motivations individuelles ayant déjà fait l'objet de plusieurs analyses[18]. Ces organisations sont de trois types principaux : les sociétés produisant du logiciel en interne, les centres de recherche, spécialement les centres de recherche publics, et les entreprises construisant des offres commerciales.

2.2.1 Participer au développement de logiciels standards, valoriser les logiciels produits en interne.

Des utilisateurs comme les entreprises sont «en concurrence», c'est-à-dire que leur travail peut a priori aussi profiter à leurs concurrents (comme pour les producteurs de logiciel) et, nous l'avons expliqué au paragraphe précédent, leurs contributions porteront principalement sur des logiciels techniques, peu orientés «métier», évoluant vite, car ce sont eux qui minimisent la parte d'avantage concurrentiel dû à la divulgation. Cette stratégie est bien illustrée par le cas «Code-Aster», logiciel développé par EDF pour modéliser des structures et des constructions : ce n'est pas le métier d'EDF, mais c'est un logiciel important (il sert notamment à étudier le vieillissement des centrales nucléaires), donc il faut s'attacher à améliorer sa sûreté. Comme le montre François Horn dans son article, la mise en libre de logiciels produits au sein d'une entreprise dont ce n'est pas le métier répond à deux objectifs principaux : il s'agit de diffuser un logiciel pour augmenter

[18]Entre autres, Lerner et Tirole [2002], Lakhani et von Hippel [2000], Foray et Zimmermann [2001], Bessen [2002], Jullien [2001].

la base d'utilisateurs, donc les conditions d'utilisation, afin d'accroître sa fiabilité, d'une part, et, si possible, d'externaliser une partie du coût de maintenance vers ces nouveaux utilisateurs, d'autre part.

Mais cela doit être considéré comme un choix stratégique impliquant des investissements. En effet, libérer un code source nécessite qu'il soit lisible, ce qui entraîne souvent des travaux de réécriture et de documentation, chiffrés par Guillaume Rousseau à environ 20 % du coût de développement[19]. Car, comme l'a fait remarquer Alain Conchon (Alcatel), «on ne veut pas non plus donner une mauvaise image de l'entreprise en publiant du code mal écrit ou peu lisible»[20]. Ensuite, cette publication va parfois à l'encontre de la culture d'entreprise, «pas toujours tournée vers le co-développement de logiciel ou même la publication des codes sources en interne» (Gérard Giraudon, INRIA), «même quand il s'agit simplement de «pseudo-libre», c'est-à-dire lorsqu'il faut que des développeurs abandonnent leur code pour qu'il puisse être adapté par d'autres développeurs en interne» (Alain Conchon). Enfin, la diffusion d'un logiciel, la création d'une communauté ne se décrètent pas, et il faut souvent faire des efforts pour faciliter l'accès, la prise en main de ces logiciels et pour les rendre compatibles avec les outils utilisés en amont ou en aval de la chaîne de production dans laquelle ils s'intègrent.

Insistons, pour finir, sur l'importance de ces contributions pour la production d'innovations dans l'industrie du logiciel. Ces utilisateurs sont, en effet, des «utilisateurs précoces», c'est-à-dire des entités exprimant des besoins qui seront peut-être ceux du marché plusieurs mois, voire plusieurs années plus tard et qui valorisent fortement la création d'une solution à de tels besoins (Hippel [von], p. 796). Donc ce sont des utilisateurs qui sont prêts à investir pour développer des solutions : on pense ici à l'exemple d'Ada 95[21], mais c'est aussi proche de lasituation d'EDF. Le choix de la licence d'utilisation du logiciel apparaît très important, pour deux raisons principales :

– si ces entreprises cherchent à diffuser un savoir et à développer les initiatives autour de leur logiciel, ces entreprises ne veulent pas non plus se voir dépossédées de leur travail et devenir dépendantes d'un producteur qui aurait récupéré le code, voir même simplement ne plus contrôler son évolution ;
– cela d'autant plus que la diffusion d'un logiciel entraîne des responsabilités légales, notamment sur les conséquences que peut avoir une mauvaise conception des logiciels, responsabilités que la licence peut permettre d'encadrer. Finalement, si la valorisation financière de relation avec les utilisateurs n'est pas l'objectif principal, elle reste coûteuse à mettre en place et catalysée, plus ou moins efficacement, par le contrat de licence que propose le producteur. On va retrouver cela dans le cas de la recherche publique.

2.2.2 Valoriser les logiciels issus de la recherche publique.

Sans doute le premier producteur de logiciel en France, la recherche publique s'interroge aussi sur ses rapports avec la production libre. Comme l'ont souligné Guillaume Rousseau (s'appuyant principalement sur son analyse des pratiques de valorisation à l'Université René Descartes) et Laurent Kott (responsable de la valorisation de l'INRIA), la publication sous une licence libre s'accorde bien avec les us et coutumes de la recherche. En effet, son principal instrument de récompense est la reconnaissance par les pairs, qui nécessite donc leur jugement, ce qui impose la publication des travaux, dont le c ?ur est aujourd'hui, dans un grand nombre de disciplines, présenté sous la forme d'un algorithme. Mais, d'un autre côté, ces institutions ont pour objectif de valoriser le travail des chercheurs, dont, lorsqu'il s'agit de programmes informatiques, elles sont propriétaires et dont elles peuvent aussi être tenues pour responsables des éventuels dysfonctionnements. Et

[19]Nous citons ici des interventions faites lors de la deuxième journée de travail de ce projet, qui sont disponibles sur http://rntl.alcove.com/workshop2/.

[20]François Horn faisant d'ailleurs remarquer que lors de la première réunion de travail, un représentant de Thalès avait confi rmé qu' «[ils avaient] des logiciels dont [ils ne servaient] plus, mais [qu'ils ne voulaient pas] montrer car il n'y pas eu assez d'investissement sur la documentation et la structuration du code».

[21]Langage dont le développement a été fi nancé par le Ministère de la défense étasunien (DoD), qui voulait disposer d'un langage de programmation objet, robuste et dont l'évolution ne soit pas tributaire de la stratégie d'une entreprise (d'où sa publication sous licence GPL)

pour elles aussi la publication en libre est coûteuse. Ainsi, «l'effort de l'INRIA aujourd'hui dans la promotion des logiciels libres est de 20 Hommes/an, auquel il faut ajouter les efforts de création de consortia : autour de CAML, autour de Scilab, etc.» (Laurent Kott)

Malgré ces difficultés, il semble que les organismes de recherche veuillent développer leurs portefeuilles de logiciels libres. Pour des raisons pragmatiques d'abord : sachant qu'il est difficile d'empêcher les chercheurs de publier les logiciels, il est sans doute plus efficace de les accompagner dans leur démarche, de leur proposer des solutions en terme de licence, de supports (serveurs) de publication qui les satisfassent. Cela permet d'abord de développer la notoriété de l'institution, parmi les centres de recherche, mais aussi les entreprises qui peuvent plus facilement essayer les logiciels. On peut aussi, mener une politique d'essaimage et de transfert de technologie, créatrice d'emplois privés et de financements pour la recherche publique. L'histoire de la bibliothèque de logiciels «Alliance» présentée par Guillaume Rousseau dans le deuxième workshop en est un très bon exemple : à ce jour, la moitié des doctorants du laboratoire initiateur d'Alliance est financée grâce à la valorisation de ces logiciels et deux entreprises de services et de développement de logiciels (qui produisent et distribuent des composants de la bibliothèque sous format propriétaire) ont été créées.

Valoriser la recherche publique grâce au logiciel libre est possible, mais demande la mise en place d'une stratégie, impose des contraintes en terme de qualité du logiciel fournit (les mêmes que pour les utilisateurs-développeurs de logiciel), de prise en compte des standards industriels dans leur conception, de développement d'entreprises de service support si cela est nécessaire et bien sûr, dans ce cas, du développement d'une politique de licence adaptée[22].
Cette politique devra, notamment, prendre en compte les caractéristiques du logiciel et du marché sur lequel il se trouve, caractéristiques qui déterminent les activités marchandes qui peuvent être développées à partir des ces logiciels libres.

2.2.3 Inventer les marchés du logiciel libre.

Avant d'étudier en quoi les logiciels libres permettent d'inventer ou, au moins, de faire évoluer des marchés en informatique, il est bon de rappeler que de nombreux contributeurs, comme IBM, SUN ou Netscape, font des offres à base de logiciels libres et contribuent à leur production pour des raisons stratégiques qui n'ont pas obligatoirement ou directement à voir avec le caractère libre des logiciels qu'ils soutiennent (même si l'on retrouve l'un des arguments de la publication sous licence libre, la création de standards ouverts, de «normes»). En même temps, ces stratégies facilitent, permettent parfois la diffusion de ce modèle, comme l'illustre l'histoire du PC.

Les stratégies opportunistes des producteurs traditionnels, l'exemple des constructeurs informatiques.

Dréan [1996] (p. 230) note que les deux moteurs concurrentiels du secteur des progiciels sont «la rivalité entre le leader de chaque sous-secteur et ses challengers» et «l'élargissement continu du champ d'application de l'informatique, qui fait apparaître de nouveaux domaines [...] où tout nouvel entrant peut espérer avoir sa chance de devenir leader à condition d'être le premier à y entrer».
Alors, lorsqu'on veut imposer un logiciel, un langage comme un standard, il peut être bon de s'intéresser aux logiciels qui l'utilisent ou qui coopèrent avec lui. Le meilleur exemple de ce type de comportement est sans

[22]Concernant ce dernier point, il existe cependant une obligation propre aux organismes publics, qui est celle de l'utilisation de contrats en langue française, rendant le choix de la GPL problématique. Constatant, avec Mélanie Clément-Fontaine, que «la loi est beaucoup plus exigeante pour les personnes morales de droit public ou chargées d'un service public» et qu'il existe une réelle diffi culté dans «la transcription de la GPL en langue et en droit français qui pose de véritables problèmes et notamment de vocabulaire», les participants ont insisté sur l'importance des efforts consacrés à cette traduction ou, pour le moins de la possibilité de disposer d'un modèle de licence commun à tous les organismes publics. Ce modèle permettrait de développer ces politiques de valorisation sur des bases juridiques solides et de créer une masse critique de production assurant sa reconnaissance par le milieu industriel et par les développeurs.

doute le travail que SUN fournit pour développer le logiciel libre Apache. Cette entreprise s'occupe de toute la partie qui concerne l'adaptation de Java à Apache. En rendant compatible le langage qu'elle a développé avec le standard du marché, elle peut espérer vendre les outils de développement Java qu'elle produit (ainsi que son expertise dans le domaine). Les producteurs sont prêts à subventionner une plate-forme pour favoriser des logiciels fonctionnant grâce à elles.
On peut estimer qu'il s'agit là de subventions croisées, où l'on subventionne un outil standard ouvert et complémentaire à son offre logiciel, comportement classique en informatique : Genthon [2000] souligne que, pour favoriser son standard (ou son produit), il faut, entre autre, «établir des alliances avec des co-producteurs» et «sponsoriser les premiers utilisateurs et les premiers produits complémentaires». Plus généralement, les challengers peuvent alors vouloir subventionner un standard concurrent au standard dominant. Le soutient que les constructeurs de machine apportent à Linux n'est sans doute pas étranger au succès de Microsoft et à sa position dominante dans le marché des PC. Surtout, ces entreprises qui dépendent d'un standard pour réaliser leur activité (comme SUN avec Apache ou les fabricants de PC et de logiciels avec le système d'exploitation) ont intérêt à ce que ce standard soit le plus ouvert possible, pour ne pas dépendre de la stratégie de l'entreprise qui le fournit, pour pouvoir diriger ses évolutions ou au moins s'assurer que les évolutions ne se font pas à son détriment. Ce type de processus de standardisation est un exemple typique de la situation où chaque acteur veut pousser sa solution, mais valorise plus l'existence d'un standard que le succès de sa proposition. Développons ce point, qui est très important pour la diffusion du Libre : les fabricants traditionnels reproduisent aujourd'hui les comportements qu'ils ont déjà eus dans le passé face aux innovations comme Unix ou les micro-ordinateurs : voyant qu'il existe une demande, ils cherchent à intégrer la nouvelle offre libre dans leur portefeuille d'offre, comme ils l'ont fait avec les innovations précédentes (Unix, micro-ordinateurs, etc.) C'est d'autant plus logique qu'ils ne maîtrisent pas un standard dominant et que, dans ce cas, ils ont tout intérêt à favoriser le développement d'un concurrent à ce standard dominant. Ce faisant, ils légitiment l'offre libre en la mettant au même niveau que les autres offres de système d'exploitation et facilitent sa diffusion. Cette attitude est illustrée dans notre rapport par l'interview du responsable d'IBM pour Linux en Europe francophone.

L'impact premier sera sans doute pour le marché des machines sous Unix, le deuxième marché en valeur après celui des micro-ordinateurs, mais aussi un marché morcelé, plus proche dans son organisation de l'informatique traditionnelle que de l'industrie des PC. Par conséquent, comme le montre Genthon [1995], cette organisation industrielle est moins efficace en ce qui concerne la production de matériel que celle du PC, à tel point que les performances matérielles sont aujourd'hui devenues aujourd'hui comparables. Les fabricants de machines Unix sont aujourd'hui en concurrence avec les fabricants de stations de travail à architecture «Intel» qui utilisent principalement Windows 2000 comme système d'exploitation (au moins sur les marchés des petits serveurs). L'arrivée des logiciels libres sur ce marché a un double impact : d'un côté, elle crée une offre d'un Unix standard, indépendant des plates-formes et elle renforce l'attractivité des systèmes Unix (qui sont le support privilégié des logiciels d'utilisation libres, même s'ils fonctionnent aussi pour la plupart sous Windows) ; d'un autre côté, elle accentue la pression concurrentielle sur les fabricants de machines sous Unix, en conduisant à l'uniformisation de l'offre dans le système d'exploitation,et parce que GNU/Linux permet de proposer un Unix fonctionnant sur les PC. Finalement, cette diffusion ne fait que renforcer, au niveau du matériel, un processus déjà en cours : celui de la diffusion de la structure ouverte de l'industrie du PC à l'ensemble de l'industrie (c'est ce que Horn [2000b] a appelé «la fusion tendancielle des réseaux»[23].

Autre type de producteur traditionnel qui voit son métier évoluer, sans pour autant qu'il soit vraiment remis en cause, celui des fabricants et éditeurs de composants, outils logiciels comme les logiciels de base de données ou les compilateurs.

[23]Voir aussi l'article de Maris Coris et Jullien [1999].

Les fabricants-éditeurs de logiciel techniques.

Dans le secteur des logiciels techniques, lorsqu'il existe un logiciel libre, il est, le plus souvent, contrôlé par une entreprise. Elle peut être à l'origine du logiciel (comme Digital Creation avec Zope ou Matra Datavision avec Open Cascade) ou avoir été crée par des personnes intégrées dans le noyau des développeurs (comme ACT avec Ada 95, Scriptics avec TCL, Cygnus avec GCC, etc.) Ces entreprises contrôlent et garantissent l'évolution du logiciel et vendent du service d'assistance à l'utilisation de leur(s) outil(s). Pour ces entreprises, la marque est liée à la marque du logiciel. Elles cherchent souvent à être reconnues comme l'entreprise d'un logiciel.

L'importance de la stratégie «libre» s'explique : ces logiciels techniques sont les logiciels à la base de toute infrastructure informatique. Ce sont donc des logiciels qui doivent être parfaitement connus de leurs utilisateurs, pour pouvoir les configurer, garantir la sécurité des installations informatiques et adapter les services demandés aux besoins de ces utilisateurs ou des organisations qui les emploient. Les exigences de qualité, d'assurance-qualité, de respect des standards prennent tout leur sens. Leurs utilisateurs valorisent particulièrement l'ouverture du code-source et leur non-appropriabilité, d'autant plus qu'ils sont aussi les mieux informés de l'existence des logiciels libres, les plus à même de les évaluer et de les adapter à leurs besoins.
C'est aussi dans ces domaines que l'utilisation de logiciels libres est la plus ancienne, notamment grâce à Internet, que les innovations venant du monde de la recherche et des utilisateurs sont les plus nombreuses, innovations favorables à la diffusion du modèle Libre. Finalement, c'est souvent pour ces logiciels que les producteurs ont le plus intérêt à choisir une stratégie libre : les marchés sont des marchés de service (ce sont des outils qu'il faut adapter aux besoins spécifiques des organisations), les utilisateurs sont capables d'évaluer la qualité technique des développements proposés par les producteurs et de proposer leurs propres contributions. C'est surtout vrai lorsqu'il existe une offre propriétaire, dominante, la diffusion en Libre permettant plus facilement aux utilisateurs d'essayer cette alternative.

Mais le métier reste assez proche de celui des constructeurs d'outils-logiciels propriétaires, comme Oracle ou Ilog : toutes ces entreprises vendent du support, le plus souvent au forfait, ou de l'adaptation de composants aux utilisateurs finaux ou à d'autres producteurs de technologies d'utilisation. Plus qu'une révolution, l'évolution vers le Libre ne fait qu'accompagner l'évolution vers la production de composants, qui augmente le besoin de certificat de qualité et surtout la garantie d'interfaces normalisées, bref de services de «mise à disposition de capacités techniques entretenues», pour reprendre la classification de Gadray [1998].
Pratiquement, ces «producteurs de composants» doivent relever un défi commercial défi souligné par Pierre Bruno, le directeur général d'Open Cascade[24] : il s'agit de définir des offres de service qui permettent de transformer une partie importante des utilisateurs de ces logiciels en clients. La spécificité de ces entreprises, par rapport aux constructeurs, aux éditeurs de logiciel ou aux entreprises de service que nous abordons dans les paragraphes suivants, étant qu'elles supportent la très grande partie des coûts de maintenance et surtout de développement des logiciels libres sur lesquels ils basent leur offre. Il s'agit alors de transformer un inconvénient (des investissements importants) en avantage commercial, soit en proposant des versions «officielles», certifiées[25], soit en instaurant des stratégies de doubles licences, c'est-à-dire en vendant la version la plus récente sous une licence propriétaire et en libérant l'ancienne afin de faire connaître le produit.

Reste qu'une entreprise comme ACT Europe, qui contrôle le compilateur Ada 95 GNAT et dont Aymeric Poulain-Maubant propose une monographie, est rentable parce qu'elle réussit à vendre la certification de son logiciel et surtout de l'assistance au développement, mais aussi parce qu'elle n'a pas eu à supporter les coûts initiaux de développement de son logiciel. à la décharge de cette entreprise (ou plutôt de son modèle économique), il faut aussi noter que ce client a financé ce développement (et la protection par la GPL) car il estimait que c'était le meilleur système pour lui garantir un produit de qualité et une indépendance vis-à-vis de son producteur. D'autre part, il est courant que le développement de logiciels propriétaires soit initié par

[24]François Horn présente cette entreprise dans son exposé.
[25]Ce qui veut dire souvent des garanties sur la stabilité, sur les délais de réponse lorsqu'un problème survient, etc.

des recherches publiques[26]. Il n'est alors pas surprenant que les stratégies qui supposent de faire supporter par le service l'ensemble des coûts de développement, plus coûteuses, illustrées par le cas d'Open Cascade (développé par François Horn), ont un retour sur investissement plus incertain, car plus long (mais c'est alors aussi le cas avec un système de licence). Mais même dans ce cas le plus défavorable, nous devons reconnaître que ce modèle économique nous semble plus cohérent et plus solide que celui des éditeurs, et notamment des éditeurs de distributions de Linux, dont l'avantage concurrentiel et la part de valeur ajoutée contrôlée semblent menacés, aussi bien par les constructeurs informatiques que par les entreprises de service.

Les «éditeurs de distributions».

Ces entreprises, Mandrakesoft, SuSE ou RedHat, se différencient des précédentes par le fait de se spécialiser dans l'assemblage de multiples logiciels, autour d'un système d'exploitation, se plaçant ainsi entre les constructeurs de machines et les constructeurs de logiciels fonctionnels. éléments sans doute fondamentaux de la diffusion de GNU/Linux et donc du phénomène libre, les producteurs de distribution semblent aussi toujours à la recherche d'un modèle pérenne.
L'exemple de Mandrakesoft en France, détaillé par Aymeric Poulain-Maubant, qui fait écho à ceux de RedHat aux états-Unis ou de Suse en Allemagne, montre qu'il n'est pas rentable, aujourd'hui, de produire et de distribuer, par les canaux habituels (librairies, revendeurs informatique), des distributions standards de GNU/Linux. La limite de ce positionnement réside dans le fait que, comme ces distributeurs s'appuient sur des produits libres, leurs produits peuvent être concurrencés, distribués par d'autres producteurs. Ils ne constituent pas une source de revenu mais simplement un produit d'appel pour les services de maintenance. Car, comme c'est le cas pour les éditeurs traditionnels comme Microsoft, la construction d'une marque de distribution, signal de la qualité des produits, est importante : c'est elle qui va leur permettre d'atteindre une partie leur public, celui des utilisateurs naïfs ou au moins non-experts, qui ne disposent, pour évaluer la qualité des offres, que de ce signal en plus de l'information que donne le prix. à partir du moment où elles auront conquis un public important, elles espèrent pouvoir développer des ressources en lui proposant des services à valeur ajoutée, augmentant ainsi leur «ARPU[27]». à cause des effets de standardisation, ces offres seront vraisemblablement limitées à quelques distributions principales, même s'il est envisageable qu'elles soient déclinées localement par plusieurs éditeurs/distributeurs.

Deux stratégies économiques émergent : soit elles spécialisent leurs distributions (RedHat et dans une moindre mesure Mandrakesoft), soit elles développent des offres de services standards, plus orientées vers le grand public (idée de clubs d'utilisateurs chez Mandrakesoft, où, contre une adhésion annuelle, sont accessibles différents services en ligne et certains logiciels). Il s'agit, dans le second cas, de standardiser, d'industrialiser le métier de constructeur d'architecture, qui comprend une forte dimension d'assurance, de garantie d'après-vente, en proposant des assemblages standards de logiciel et des services standards d'assistance.

L'orientation vers les services à façon a comme conséquence de placer des éditeurs en concurrence avec les entreprises de services, que nous abordons ci-après, en s'inspirant fortement du modèle des fabricants de composant, supposant implicitement qu'il y a un avantage concurrentiel, en terme de compétences maîtrisées, à développer une distribution. Et il n'est pas sûr que le fait de proposer, comme base de son service, une suite de logiciels que l'on contrôle, soit dans ce cas, un avantage. En effet, les clients peuvent craindre d'être dépendant de l'éditeur de la distribution pour les développements futurs, car il existe un risque que, pour augmenter leurs marges, ces producteurs proposent des améliorations non parfaitement compatibles, afin de différencier leur offre, mais aussi de rendre plus captifs leurs clients. Cependant, l'ouverture des logiciels devrait permettre de garantir une compatibilité suffisante entre les versions, faisant ressembler la concurrence à une concurrence entre producteurs de biens compatibles, donc permettant la survie de plusieurs offres (surtout

[26] Pour l'exemple, Internet, ou, plus récemment, Unix BSD, repris par Apple avec MacOS X et par Microsoft pour construire certains éléments des Windows.

[27] «Average Revenu Per User», terme utilisé principalement dans les télécommunications, qui permet d'évaluer la rentabilité d'une entreprise en se basant sur les revenus moyens générés par un utilisateur.

que les principaux «éditeurs» s'impliquent dans des instances de normalisation, comme le «Free Standard Group» [28]). Celle vers les clubs d'utilisateurs n'a pas, pour l'instant, permis à Mandrakesoft de dégager des profits, même si la jeunesse de l'initiative (moins de six mois) n'autorise certainement pas de conclure sur sa pertinence. Finalement, les rapprochements de SuSE et d'IBM, ou de Mandrakesoft et de Sun (via la diffusion de Star Office), montrent que les constructeurs informatiques commencent à investir ce domaine, qu'ils connaissent déjà étant distributeurs de versions d'Unix, afin de développer la composante service de leurs revenus. Cela risque de réduire d'autant l'espace disponible pour un éditeur indépendant.

Considérant la tradition européenne du service, considérant aussi les caractéristiques du Libre, il n'est pas étonnant de constater que la majorité des entreprises dont comme stratégie de se concentrer sur des services basés sur les logiciels libres[29]. Proposent-elles un nouveau modèle, quelle est leur spécificité vis-à-vis des entreprises de service classiques, les SSII ?

Les sociétés de service en logiciels libres.

Ces «SSLL[30]» se positionnent sur des marchés d'expertise où il s'agit de développer, à la place du client, des solutions individualisées et de maintenir ces solutions. On est proche de ce que Gadray [1998] appelle la «mise à disposition de capacités humaines», en ce sens que ce qui fait la particularité de ces entreprises est qu'elles regroupent en leur sein une équipe de spécialistes de différents logiciels. Leur renommée se construit parmi les utilisateurs développeurs, designers ou sophistiqués, en même temps que leur expertise par des contributions diverses à de nombreux logiciels. Les logiciels libres sont utilisés parce qu'ils permettent de créer une solution plus souple, indépendante des éditeurs et des constructeurs et une relation de service plus équilibrée, à plus long terme[31].

Elles doivent intégrer deux impératifs, contradictoires : il faut être capable de comprendre des besoins qui s'expriment en terme de métier, ce qui signifie pouvoir s'appuyer sur des bibliothèques de fonctionnalités, donc de logiciels, tout en ayant une expertise sur certains logiciels, seule capable de créer une valeur ajoutée à l'utilisation de logiciels libres.
Résoudre ce dilemme peut se faire de trois manières : soit les entreprises du Libre se concentrent, afin de conjuguer expertise et pluralité sous une même structure, soit elles réussissent à constituer des regroupements de spécialistes de technologies-logiciels, soit elles deviennent des sous-traitants des SSII classiques.

Cette dernière éventualité est évidemment la moins favorable pour les sociétés de services autour des logiciels libres : elles risquent d'être intégrées verticalement, ou au moins d'être en situation de dépendance vis-à-vis de ces dernières, dans un schéma qui ne serait pas sans rappeler celui de la construction, avec des «Bouygues» du service informatique faisant travailler une myriade de petites entreprises plus ou moins spécialisées.

Mais si les deux premières alternatives offrent plus de perspectives, elles ne sont pas non plus sans poser de problème.
Dans le premier cas, il y a un risque de perdre ce qui est apparu comme une des spécificités des SSLL, la gestion des ressources humaines : les employés de ces sociétés bénéficient d'une grande autonomie dans leur travail, il est considéré comme normal qu'ils aient du temps pour contribuer à des projets libres et le recrutement par cooptation, lorsqu'il s'agit de personnes avec qui on a l'habitude de travailler sur ces mêmes

[28] http://www.freestandards.org/news.php?id=35

[29] La présentation de Marie Coris s'intéresse à ces entreprises.

[30] Ce terme SSLL, s'il a connu un certain succès (lors de la première réunion, il était revendiqué par ces entreprises comme signe distinctif), semble, au dire des participants, avoir perdu de sa clarté, sans doute parce qu'il est devenu une marque déposée par l'une d'entre elles. Nous le conservons cependant, car il est aujourd'hui couramment utilisé, notamment dans la presse informatique, pour désigner ces entreprises.

[31] Un des points qui ressort du traitement des données de notre enquête (analyse de donnée) est qu'une des caractéristiques des SSLL est qu'elles privilégient la distribution GNU/Linux Debian pour construire leurs offres de service. C'est la seule distribution majeure qui n'est pas contrôlée par une entreprise

projets libres, est très courant[32]. La concentration des entreprises, augmentant leur taille, risque de rendre plus difficile cette gestion «familiale», où la collaboration se fait de façon largement informelle et où les frontières entre travail et activité personnelle ne sont pas toujours clairement établies. Enfin, ces entreprises risquent de se heurter aux «éditeurs» de distributions, voir même aux constructeurs de machines, dont le métier semble justement s'orienter vers l'assemblage de logiciels dans une distribution déclinée suivant les besoins des clients.
Dans le deuxième, se posent évidemment les problèmes de la coordination entre spécialistes de logiciels, de la localisation des compétences de traduction de besoins fonctionnels en terme de solution logiciel et de la gestion de la construction des ces solutions. Enfin, la spécialisation autour d'un logiciel n'est efficace si cela crée une valeur ajoutée, si l'on est capable de la vendre. L'exemple des «producteurs de composants», tels Open Cascade ou ACT montre que ce modèle n'est pas non plus stabilisé.

2.2.4 Quel paysage économique ?

Au-delà de la classification, cette analyse montre que les différentes stratégies s'hybrident, encore à la recherche du modèle économique le plus efficace, le plus rentable. à partir de là, on se rend bien compte que proposer une analyse prospective de l'évolution du paysage économique est un exercice difficile, tant les éléments à prendre en compte son nombreux. Cependant, il semble que la prolongation des comportements stratégiques constatés permet de dégager deux types de paysage, assez différents, dont le choix va être, comme par le passé, conditionné au positionnement stratégique des constructeurs de machines et aux préférences des utilisateurs, principalement des utilisateurs industriels.

Les constructeurs de machines doivent fournir avec celles-ci un système d'exploitation. La logique industrielle «Unix» veut que ce soit ces constructeurs qui maîtrisent le développement du système, quand la logique «PC» le délègue à un fournisseur externe. Actuellement, les constructeurs ont tendance à reproduire ces mêmes pratiques dans leur utilisation des systèmes d'exploitation libre : quand IBM ou HP proposent plusieurs distributions (HP allant même jusqu'à choisir la distribution Debian comme distribution de référence), les constructeurs de PC comme Dell nouent des partenariats avec des producteurs de distribution (RedHat pour Dell). Si ces choix étaient prévisibles : il est plus facile de faire évoluer les «routines» organisationnelles et commerciales que de les révolutionner, ils proposent deux types d'organisation industrielle.

Si, dans une organisation industrielle proche de celle des offres basées sur Unix, les constructeurs réintégraient une partie du contrôle sur le système d'exploitation, ce serait pour développer les offres de service. Ils se placeraient alors en position de concurrence avec les entreprises de service comme Alcôve, en faisant disparaître les éditeurs de distributions libres. En même temps, l'utilisation d'un système d'exploitation qu'ils ne contrôleraient pas complètement rendra toujours possible l'utilisation d'un ou plusieurs fournisseurs de machines et d'une entreprise de service pour assurer leur cohabitation. Si c'est la «culture PC» qui s'impose, les constructeurs risquent d'être relégués au rôle de simple assembleur de composants, les revenus de services étant captés par les fabricants de distributions, RedHat, SuSE ou Mandrake, qui seraient alors les entreprises dominantes de cette nouvelle période industrielle. L'organisation industrielle serait proche de celle qu'on trouve actuellement dans l'industrie du PC, où les marges sont extrêmement réduites pour les assembleurs de machines et très confortables pour les éditeurs de système d'exploitation et de logiciels. Mais l'ouverture, qui facilite la compatibilité, devrait permettre la survie de plusieurs fournisseurs, ce qui garantirait une certaine concurrence entre les distributions. On serait dans le cas où le système d'exploitation se rapprocherait d'un composant, qui serait installé par les fabricants des machines architecturées suivant les demandes des clients.

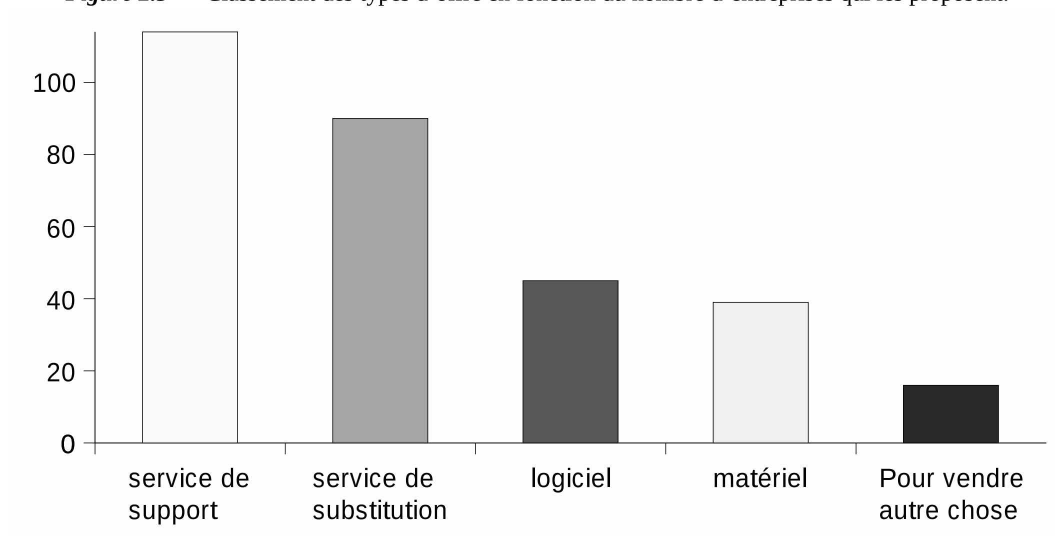
Les constructeurs de machines seront en partie responsables du sens que prendra l'organisation industrielle, car ce sont eux qui sont actuellement en position de force vis-à-vis des éditeurs de distribution, à cause de leur taille. Mais cette évolution dépendra aussi des entreprises utilisatrices, puisque ce sont elles qui vont

[32] Certaines sociétés, comme Easter Eggs (http://www.easter-eggs.com/), proposant même des structures industrielles particulières, où la société est possédée à égalité par les différents employés.

sélectionner les offres.
Le premier système devrait être favorisé par les utilisateurs-développeurs : c'est celui qui leur garantit le meilleur contrôle sur les évolutions des logiciels qu'ils utilisent, où le degré de normalisation est le plus fort et où le moins de choix techniques sont laissés au contrôle d'une entreprise. Pour les mêmes raisons, cette organisation devrait être privilégiée par les entreprises qui choisissent les offres libres à cause des garanties sur la pérennité des logiciels et sur leur interopérabilité. Mais c'est aussi celle qui demande le plus d'investissement de la part des clients car il faut qu'ils soient capables de suivre l'évolution des standards, ou qu'ils confient ce suivi à une entreprise de service.
Le second système demande moins d'efforts aux clients pour adapter leurs «routines» : on est dans une organisation plus proche de celle du monde des PC, avec des fournisseurs-éditeurs de systèmes d'exploitation, dont les marques sont connues. Ce système serait sans doute plus lisible pour les utilisateurs naïfs, qui auraient à choisir entre différents produits et non pas à spécifier la façon dont ils veulent construire leur machine.

Suivant l'objectif recherché par les utilisateurs, l'un ou l'autre des systèmes semble meilleur. On peut imaginer qu'ils cohabitent : le premier système pour les applications les plus techniques, et notamment pour les serveurs (donc dans la première phase de la diffusion), le second lorsque le choix est moins technique, lorsqu'il s'agit de distribuer le même système sur un grand nombre de postes, ou lorsque les clients ont moins de compétences techniques (nous pensons, par exemple aux petites entreprises).

***Figure 2.5* —** Classement des types d'offre en fonction du nombre d'entreprises qui les proposent.



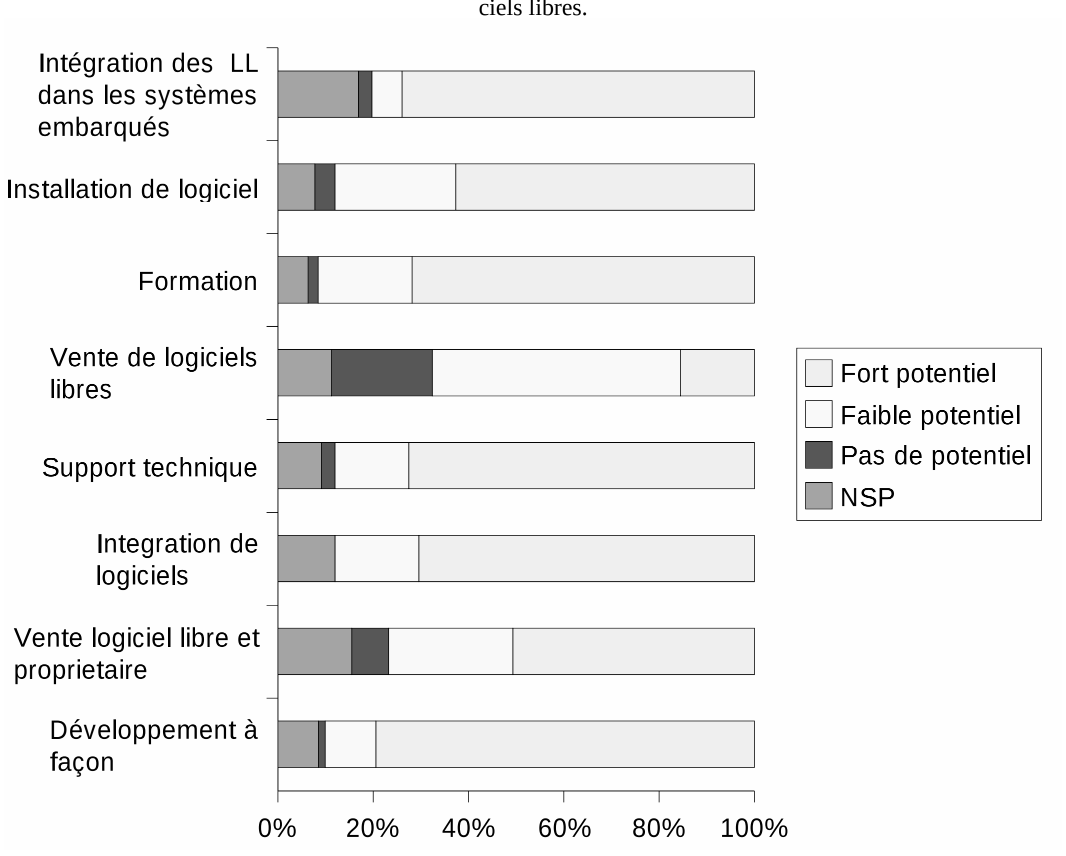
Ce qui est d'ors et déjà certain c'est qu'«il a un seul moyen de gagner de l'argent, les services. Parfois l'aspect service est trop limité et il n'est pas évident qu'il y ait possibilité d'existence d'une activité marchande. C'est finalement toujours la même question de l'articulation d'activités marchandes et de logiciels libres.» (François Horn).

Cette articulation se fait dans l'élaboration contractuelle des relations, que ce soit au niveau des licences des logiciels dont Laure Muselli analyse ce qu'elles signifiaient sur le positionnement stratégique des entreprises qui les utilisent, ou des engagements de service. On perçoit alors toute l'importance des questions juridiques dans l'appréhension de l'écologie du Libre.

2.3 Quel cadre juridique de la production de logiciel, aujourd'hui et demain ?

Le logiciel, qui est devenu une valeur économique importante, fait partie des créations immatérielles protégeables par le droit de la propriété intellectuelle.

Figure 2.6 — Estimation par les entreprises du potentiel des activités commerciales utilisant des logiciels libres.



Le législateur français, dès 1968[33], a exclu la brevetabilité des logiciels sans pour autant désigner le régime applicable. Ce sont donc les tribunaux qui, les premiers, ont retenu la possibilité de protéger le logiciel par le **droit d'auteur**[34]. La loi du 3 janvier 1985[35] a entériné ce choix. Toutefois, l'aspect technique du logiciel a été pris en considération, ainsi bon nombre de dispositions applicables aux logiciels dérogent au droit commun des auteurs.

Parmi ces règles dérogatoires, on retiendra la dévolution automatique des droits patrimoniaux à l'employeur «sur les logiciels et leurs documentations créés par un ou plusieurs employés dans l'exercice de leurs fonctions ou d'après les instructions de leur employeur» (article L113-9 du Code de la Propriété Intellectuelle) ; ou encore, la disparition de l'exception de copie privée qui est remplacée par l'exception de copie de sauvegarde. L'utilisateur bénéficie de droits parfois plus étendus que ceux prévus par le régime commun : ainsi, à certaines conditions, il peut reproduire, traduire voire adapter le logiciel (article L 122-6 CPI). Par ailleurs, la protection de l'intégrité de l'œuvre n'a plus la même force[36]. En effet, le logiciel étant perçu avant tout comme un outil, l'utilisateur légitime peut corriger ou décompiler le code. Enfin, les droits

[33]L. n°68-1, 2 janvier 1968 : *J.O. 3 janv., p.13*. La même solution a été retenue par la Convention de Munich du 5 octobre 1973, article 52.

[34]V. M. Vivant et C. Le Stanc et alii, *Lamy droit de l'informatique* 2002, n° 151.

[35]L. n°85-600 du 3 juillet 1985 : *J.O. 4 juill. 1985, p. 7495*. Afi n d'harmoniser les régimes de protection des logiciels des pays de l'Union, une Directive fut adoptée par les autorités communautaires le 14 mai 1991 (*Dir. Cons. CE n°91/250, 14 mai 1991, JOCE 17 mai 1991, n° L 122, p. 42*). Elle fut transposée en France par la loi du 10 mai 1994 (*L. 94-361, 10 mai 1994, J.O. 11 mai 1994*).

[36]A. et H.-J. Lucas, *Traité de la propriété littéraire et artistique*, 2 éd. Litec, 2001, n°419.

de retrait[37] et de repentir[38] n'ont pas été retenus (article L 121-7 CPI)[39].

L'utilisation permise d'un logiciel fait l'objet d'une description dans un contrat qui prend souvent la forme d'une **licence** s'agissant des progiciels[40]. S'il convient que la licence respecte la législation applicable au logiciel, celle-ci offre de nombreuses possibilités au donneur de licence pour aménager les conditions d'utilisation du logiciel. En effet, les dispositions légales sont souvent supplétives, autrement dit elles s'appliquent en l'absence de stipulation contraire prévue dans le contrat. Par exemple, selon l'article L 122-6-1 du Code de la Propriété Intellectuelle, certains actes, tels que faire des reproductions du logiciel, le traduire ou l'adapter quand c'est nécessaire pour l'utilisation du logiciel conformément à sa destination, ne sont pas soumis à autorisation. Toutefois, la licence peut prévoir que seul l'auteur pourra corriger les erreurs et déterminer les modalités particulières auxquelles seront soumis de tels actes.

Aussi, le donneur de licence dispose d'une certaine marge de liberté pour définir les autorisations qu'il confère aux utilisateurs. Les licences de logiciels libres sont un parfait exemple de cette prise de liberté.

Avant même ces licences, il existait une «pratique du libre». Sans elle la face d'Internet serait certainement différente. La plupart des briques d'Internet sont en effet des logiciels libres : Sendmail, Postfix pour le courrier électronique, le projet BSD pour ses apports majeurs aux serveurs du Net, Bind pour la gestion des noms de domaines ou encore Apache pour les serveurs web. Les normes et les protocoles ont été mis à disposition avec un appel à contribuer à leur évolution[41]. Ce fut le cas, par exemple, pour le protocole TCP-IP. Ceci a eu lieu au tout début des années 70, alors même qu'il existait d'importants organismes de normalisation (ISO, ANSI, AFNOR, IEEE) et de gros constructeurs qui développaient et prêchaient pour leurs propres systèmes de réseaux (Digital, IBM, France Télécom)[42]. Il s'agissait de favoriser la circulation et donc l'échange des connaissances et de créer des standards maîtrisables par tous. Ainsi ces logiciels sont pour la plupart soumis à des licences qui offrent expressément aux utilisateurs les libertés de copier, modifier et diffuser le logiciel (modifié ou non). Les premières licences de logiciels libres ont vu le jour aux Etats-Unis dans les années 80 sur l'initiative d'organismes de recherche[43] et de personnes privées[44], qui développaient eux-mêmes des logiciels.

Le besoin de formaliser, sous la forme de licences, une pratique communément partagée dans le milieu de la recherche vient en réaction à une tendance réservataire des marchés du logiciel.

En effet, la valorisation marchande des logiciels a reposé et repose encore essentiellement sur un système d'appropriation autorisant l'usage d'un logiciel, le plus souvent moyennant une rémunération : le titulaire des droits se réserve l'exclusivité de l'exploitation des versions futures en interdisant la modification du logiciel. S'il existe une exception légale autorisant l'utilisateur légitime[45] de modifier le logiciel pour corriger

[37]Le droit de retrait permet à l'auteur de «retirer entièrement l'œuvre du commerce, c'est-à-dire faire (de) faire cesser l'exploitation». P. Sirinelli, *Propriété littéraire et artistique et droits voisins*, 1992 Mémentos Dalloz, p. 56

[38]Le droit de repentir permet à l'auteur de «remanier l'œuvre». P. Sirinelli, *Propriété littéraire et artistique et droits voisins*, 1992 Mémentos Dalloz, p. 56.

[39]Voir notamment sur le régime de protection applicable au logiciel : C. Colombet, *Propriété littéraire et artistique et droits voisins*, Précis Dalloz, 1999, p. 85 et s., P-Y Gautier, *Propriété littéraire et artistique*, 2001 ; M. Vivant (sous la dir.), *les créations immatérielles et le droit*, Ellipses, 1997, p. 50 ; M. Vivant et C. Le Stanc et alii : *Lamy droit de l'informatique* 2002, n°140 et s.

[40]Le progiciel est un programme «standard» contrairement au logiciel «spécifi que» qui est conçu pour un utilisateur déterminé.

[41]La publication est alors faite sous la forme d'une RFC (*Request For Comment*).

[42]Voir, dans le présent rapport, l'étude d'Yves Rougy intitulée «Les usages des licences libres» qui décrit notamment les domaines techniques dans lesquels on trouve de nombreux logiciels diffusés sous licences libres.

[43]Notamment, l'Université de Berkeley en Californie ou le consortium X du MIT.

[44]Ainsi, Richard M. Stallman est-il à l'origine du projet GNU.

[45]L'utilisateur légitime est celui qui a reçu l'autorisation de faire usage du logiciel. par opposition à l'utilisateur d'une version piratée.

des erreurs ou permettre l'interopérabilité[46], celle-ci est conditionnée. Les modifications sont permises uniquement dans les limites de la destination de l'œuvre[47] et si l'utilisateur ne dispose pas d'autres moyens. L'auteur peut également décider de se réserver le droit de procéder aux corrections. Enfin, ces dispositions ne doivent pas être interprétées comme pouvant permettre des atteintes à l'exploitation de l'œuvre. Il est dont aisé de se prévaloir de toutes ces limites pour réduire d'autant la marge de manœuvre de l'utilisateur qui n'a alors aucun contrôle sur son outil. En toute logique, l'accès au code source lui est interdit. Ainsi la stratégie de ces licences consiste en la réservation, pour le seul titulaire des droits de propriété intellectuelle, de la faculté d'intervenir sur le code du logiciel en rendant inaccessible leur code source et en interdisant de modifier le progiciel[48]. Si l'utilisateur moyen n'a pas conscience de ces restrictions, un utilisateur avisé bute régulièrement devant de telles contraintes. Aussi n'y avait-il pas de meilleure réponse au système des licences de logiciels «réservés» que de créer des licences de logiciels «libres» afin de permettre l'échange des outils. De fait, il se créa une «communauté du libre» constituée principalement de développeurs qui, grâce aux libertés ainsi conférées, ont pu utiliser pleinement les logiciels et le cas échéant, contribuer à leur amélioration. Aujourd'hui, le mouvement a pris une dimension internationale et s'est étendu au-delà du cercle de la «communauté du libre». Les logiciels ainsi diffusés sont entrés dans le circuit économique : de plus en plus d'entreprises sont, cumulativement ou non, utilisatrices de ces logiciels, éditrices[49], ou prestataires de services en informatique libre. Dès lors, il n'est plus possible d'envisager les logiciels libres comme une initiative marginale sans réelle portée économique. L'œuvre qu'est le logiciel ne constitue pas dans ce système une valeur marchande, puisque seuls les services[50] font l'objet de transactions.

Prévues pour prendre le contre-pied du copyright américain[51] ces licences semblent tout à fait adaptées au droit d'auteur français. Les aspects communautaire et altruiste du mouvement libre ont conduit parfois à penser qu'il s'agissait purement et simplement d'une remise en cause des droits de propriété intellectuelle. Dès lors, les licences libres ne pouvaient pas être valables en France. Pourtant, si elles constituent un véritable pied de nez à l'exploitation commerciale de la création, elles respectent les prérogatives de l'auteur. Ainsi, s'agissant du droit moral, qui en matière de logiciel est limité[52] les licences libres ne contreviennent pas au droit de divulgation appartenant à l'auteur car elles caractérisent par définition sa volonté de divulguer sa création. De même, les licences libres ne méconnaissent pas le droit de paternité étant donné qu'il n'est pas interdit de signer son propre développement. Il est parfois même exigé d'indiquer clairement la modification que l'on a faite afin qu'elle ne soit pas attribuée faussement à un autre. Quant aux droits patrimoniaux, si leur exploitation peut surprendre, rien ne l'interdit. En effet, l'article L. 122-6 du Code de la propriété intellectuelle dispose que l'auteur a le droit d'autoriser «la reproduction permanente ou provisoire d'un logiciel en tout ou partie, par tout moyen, et sous toute forme ; La traduction, l'adaptation, l'arrangement ou toute autre modification d'un logiciel et la reproduction du logiciel en résultant, la mise sur le marché à titre onéreux ou gratuit». Autrement dit, l'auteur peut très bien décider d'autoriser la reproduction en tout ou partie de son logiciel, quel qu'en soit le moyen ou la forme, la modification, la diffusion et ceci gratuitement.

[46]Selon la Directive CEE du 14 mai 1991 concernant la protection juridique des programmes d'ordinateur, l'interopérabilité est la «capacité d'échanger des informations et d'utiliser mutuellement les informations échanger» ; cité par M. Vivant et C. Le Stanc, *Lamy droit de l'informatique*, 2002, p.1934.

[47]Le terme «œuvre» est employé, dans ce développement, au sens du droit d'auteur : «la création de l'esprit empreinte d'originalité qui, comme telle, donne prise aux droits d'auteur» (Gérard Cornu, vocabulaire juridique, association Henri Capitant, éd. Puf, 1997).

[48]Les pratiques les nomment volontiers licences propriétaires pour les distinguer des licences libres. Afi n d'éviter la confusion qui consisterait à croire que seuls les logiciels sous licences propriétaires sont protégeables par le droit d'auteur à l'exclusion des logiciels libres, nous opterons pour l'expression «licences non-libres» ou licences de logiciels «réservés».

[49]éditrice non pas au sens du Code de la propriété intellectuelle mais au sens commun c'est-à-dire que des entreprises permettent le développement de logiciels libres.

[50]Les services proposés sont pour les plus classiques de l'assistance technique, de l'ingénierie, mais aussi de la garantie, ou des prestations en matière de sécurité ?

[51]Ainsi est née l'expression «copyleft» traduite par son auteur, Richard M. Stallman, par «gauche d'auteur».

[52]En effet, «sauf stipulation contraire, l'auteur ne peut pas s'opposer à la modifi cation du logiciel» (article L121-7 CPI), il ne dispose pas de droit de repentir ni de retrait. Il reste à l'auteur le droit de paternité et le droit de divulgation.

En somme, il peut diffuser logiciel sous licence libre[53].

Le paysage juridique du logiciel va peut-être changer avec la reconnaissance d'une solution jadis écartée : la **brevetabilité des logiciels**. Les associations pour la promotion des logiciels libres mettent en garde contre les effets pervers d'un tel changement[54]. La question ne manque pas d'interpeller tant les économistes[55] que les juristes[56].

En effet, d'un côté il faudrait définir les critères de protection des inventions brevetables, en l'espèce le logiciel, ou encore traiter la question du cumul de protections (par le régime des brevets et du droit d'auteur) de l'autre, il faudrait mesurer l'impact de la reconnaissance de la brevetabilité des logiciels dans l'équilibre qui existe entre les développeurs. Ces derniers ont des profils parfois très différents[57]. Ainsi, quand bien même il est envisageable de créer des licences libres de brevet de logiciel, quelles seront en pratique les chances des développeurs de logiciels libres face aux grands groupes désireux de garder le contrôle du marché de l'informatique ?

Pour l'heure, il s'agit d'analyser les licences de logiciels libres au regard du droit actuel et de leurs usages tout en gardant à l'esprit qu'à l'instar de l'économie, le droit et l'informatique sont en continuelle mutation.

[53]Voir, pour une analyse plus détaillée des clauses des différentes licences libres notre étude intitulée «les licences de logiciels libres - étude juridique» du présent rapport.

[54]Exemple : la position résumée de l'AFUL (juillet 2001) http://www.aful.org. V° également la pétition contre les brevets logiciels et pour la liberté de programmer en Europe, http://petition.eurolinux.org/.

[55]Par exemple : Y. Dupuis et O. Tardieu, *Les brevets superflus en matière de logiciel*, rapport 2001 école des Mines.

[56]Voir L'étude proposée par Franck Macrez intitulée «Brevets concernant un programme d'ordinateur : l'étendue de la protection» dans le présent rapport.

[57]Il n'y a, en effet, aucun rapport entre une entreprise multinationale, un laboratoire de recherche ou particulier qui développent des logiciels. Pourtant ils sont soumis au même titre au droit. Il n'est dès lors pas diffi cile d'imaginer combien la reconnaissance de la brevetabilité des logiciels ne représente pas du tout la même chose pour les uns et les autres.

Première partie

Enquêtes sur l’évolution de l’industrie informatique.

Les constructeurs : l’exemple d’IBM.

AYMERIC POULAIN-MAUBANT,
Jipo Interactive, Aymeric.PoulainMaubant@jipo.com.

– Personne interviewée : Marc Joly, Directeur Linux pour la France, la Belgique et le Luxembourg
– et Site de référence : http://ibm.com/linux/

3.1 Quelques rappels.

En un peu moins de deux ans, et alors que la communauté du Libre l'appelait de ses vœux —comme elle peut l'appeler pour tous les grands noms de l'informatique matérielle et/ou logicielle—, IBM a construit une stratégie et une offre Libre séduisantes.

Au milieu de l'année 2001, il était déjà possible à quelques privilégiés de se connecter via Internet sur des serveurs IBM de la classe zSeries tournant sous Linux, de manière à tester les divers applicatifs qui y avaient été portés. Cette phase de test achevée, IBM annonçait le lancement de plusieurs *mainframes* tournant exclusivement sous Linux. Dans le même temps, la grande maison lançait une campagne de publicité offensive à la télévision, première de ce genre à mettre le pingouin Tux devant les yeux de la ménagère de moins de 50 ans. La particularité de cette campagne était aussi de ne pas renier les autres choix de systèmes d'exploitation et/ou de logiciels ; partie intégrante de la stratégie d'IBM vis-à-vis du Libre, le message est que Linux et les logiciels libres permet de consolider les investissements (reconnus coûteux), en serveurs haut de gamme. Pour autant, aussi bien les grands comptes que les petites entreprises sont visées, à travers la politique e-business également activé chez IBM.

IBM estime que le marché existe, surtout avec des entreprises de plus en plus confrontées à des problèmes de réduction de coûts majeurs. Les divers matériels proposés (classes iSeries pour les PME/I, ou zSeries pour les plus gros consommateurs de ressources informatiques) avec solutions Linux sont l'équivalent de 15 à plusieurs centaines de serveurs classiques. De quoi attirer l'attention aussi bien de ceux qui cherchent à s'équiper que de ceux qui cherchent à amortir leurs investissements passés.

Pour obtenir ces résultats, et soutenir une politique orientée Libre volontariste, IBM aura investi 1 milliard de dollars en 2001, principalement en R&D dans le développement de ses logiciels et matériels sous Linux. Cet investissement a permis à IBM d'annoncer début 2002 des *mainframes* uniquement sous Linux, alors qu'en 2001 Linux était encore proposé en option, ce qui signifie beaucoup dans la confiance qu'IBM peut avoir dans ces/ses solutions. Et si IBM a principalement choisi la classe eServer zSeries pour ce faire, c'est que ces serveurs avaient été les seuls à enregistrer un taux de croissance au-dessus de la moyenne du secteur lors de pas moins de cinq trimestres consécutifs.

Si ces annonces et cette stratégie tombe bien avec la demande des entreprises de consolider leurs investissements, elle correspond également avec la tendance du retour vers les ordinateurs centraux. En effet, mettre en place plusieurs serveurs matériellement séparés serait une solution finalement assez chers. Le vent semble donc actuellement porter assez bien les ambitions Linux et logiciel libre de *Big Blue*.

Témoin de cette forte volonté d'IBM, Marc Joly a été nommé dès janvier 2001 directeur Linux pour la France, la Belgique et le Luxembourg[1] . Nous avons recueilli ses propos.

[1] Diplômé de l'École Supérieure de Commerce de Lille, il a rejoint IBM après un passage dans le secteur bancaire, et est devenu en 1996 manager pour la France, la Belgique et le Luxembourg des solutions “terminaux points de vente”. Il prend en 1999 la direction du développement des solutions e-CRM pour l'ensemble de l'Europe.

3.2 Stratégie.

RNTL : *De plus en plus d'entreprises choisissent Linux et les logiciels libres comme solution alternative. IBM a fait quelques annonces majeures sur ce marché depuis quelques mois, et rappelle que ses clients cherchent avant tout à consolider leurs investissements matériels passés, et en particulier réduire le nombre de serveurs qu'elles maintiennent pour en améliorer la gestion quotidienne et réduire les coûts. Quelle est la stratégie actuelle d'IBM en matière de logiciels libres ?*

MARC JOLY : Elle est fondée principalement sur Linux. Tout le monde peut s'accorder à constater que le marché investit rapidement Linux et les solutions logiciel libre, alors qu'il y a encore deux ans, Linux était encore du ressort des enthousiastes, et de premiers adopteurs volontaristes. En un temps relativement court, Linux a gagné ses lettres de noblesse, notamment dans les applications orientées Internet, ou dans les grands *clusters* utilisés par la communauté scientifique. Il a investi ensuite des secteurs critiques, comme l'e-commerce ou les applications bancaires. IBM accompagne donc ce mouvement en proposant le même niveau de support que nous associons normalement à toutes nos solutions. Accompagne et pousse, également, car nous pensons offrir tant à nos clients, à nos partenaires qu'à la communauté du Libre, des solutions innovantes et à de pointe.

L'offre d'IBM pour Linux est décomposée selon les trois facettes que sont : les serveurs, les logiciels, les services. Notre stratégie peut dès lors être décrite comme suit :

1. Créer une ligne de produits de pointe pour y faire tourner de manière optimale Linux et des applicatifs libres, et y associer les services nécessaires pour développer et déployer ces applications,
2. Faire en sorte que tous nos systèmes supportent en totalité Linux, co-existent avec Linux, ou sont compatibles avec Linux,
3. Développer des partenariats avec la communauté des développeurs de logiciels libres avec comme objectif l'amélioration constante de Linux.

RNTL : *Pourquoi Linux, et pas FreeBSD ou un autre système d'exploitation, ou plusieurs systèmes ?*

MARC JOLY : Linux est clairement un consensus, issu d'une observation simple de “parts de marché”. Comme toute entreprise, IBM a le devoir de survie. Et le marché est ce qui fait un sens pour le client. En l'occurrence, Linux est apparu comme le système d'exploitation qui faisait sens.

Par ailleurs, Linux est très fort pour tout ce qui est support de matériels divers, d'infrastructures diverses. C'est aussi la force d'IBM ; nous pouvons dire que *nous sommes leader en infrastructures matérielles, logicielles et services*. Les deux s'associent donc très bien.

Je pourrais également répondre à la question “pourquoi Linux, logiciel libre” ? Avoir le code source de Linux permet d'avoir un code source bien meilleur et bien plus stable que toute autre solution. La flexibilité et la liberté associée aux logiciels libres permet bien sûr de ne plus se limiter à un seul système d'exploitation, de ne plus être enfermé dans des solutions propriétaires et de réaliser des économies que ne peut pas offrir le propriétaire. On peut ainsi :

– gérer des systèmes et des données avec des applicatifs parfaitement adaptés aux besoins et aux contraintes

– réduire les coûts et optimiser les charges sur des serveurs[2]
– offrir aux équipes techniques de nos clients des environnements logiciels adaptés, adaptables et compréhensibles

RNTL : *Cette stratégie Linux d'IBM est-elle vraiment originale ? Quelle est la valeur ajoutée perçue par vos clients ?*

MARC JOLY : Nous n'avons certes pas créé quelque chose de complètement nouveau. Nous avons simplement associé Linux à nos architectures IBM déjà existantes. En aucun cas, du reste, Linux n'est appelé à remplacer les systèmes d'exploitation déjà en vigueur chez IBM. Il faut plutôt l'envisager comme un complément. Compléter l'offre déjà proposée à nos clients, mais surtout —et c'est là toute la simplicité de notre approche— les aider à utiliser au mieux leur infrastructure déjà existante.

Linux nous permet de répondre aux préoccupations majeures de nos clients actuellement : réduire le coût de possession (TCO), obtenir un meilleur taux de service et une meilleure réactivité de leur architecture.

Si l'on prend par exemple le cas d'une ferme de serveurs distribués, le délai actuel pour ajouter un serveur supplémentaire est de de quelques heures à plusieurs jours. Jusqu'à ce que nous présentions notre offre zSeries Linux[3], qui permet d'ouvrir des serveurs virtuels Linux en seulement trois minutes, et ajouter de nouveaux processeurs en moins de 24 heures. Taux de service et réactivité sont dès lors drastiquement améliorés. Ces solutions flexibles accompagnent de près les besoins incessants, et très rapides, d'accroissement des affaires de nos clients.

RNTL : *Pour l'instant, vous n'avez décrit que des solutions pour des grands comptes ou des PME. Qu'en est-il de l'offre destinée aux particuliers ?*

MARC JOLY : La stratégie d'IBM en matière de logiciels libres est pour l'instant uniquement destinée effectivement aux serveurs d'entreprises, dans le but de simplifier et consolider les architectures de nos clients. Quand les applicatifs destinés habituellement au grand public auront un écho suffisant dans la communauté du Libre, le marché grand public se dirigera alors peut-être plus clairement vers ces solutions. Mais nous n'en sommes pas encore là. C'est le marché qui décidera, donc, si nous nous engageons également fortement dans le Libre pour les particuliers.

IBM suit ces évolutions de près. Pour en revenir au marché des entreprises, il est mature pour tous les applicatifs orientés sur les serveurs et les infrastructures, comme les proxies, les DMZ, mais pas encore pour des applicatifs de type ERP ou CRM. En l'an 2002, ce sont les architectures orientées Web, les serveurs de messagerie, d'impression, les applications à fortes contraintes de fiabilité et de sécurité qui sont les domaines de prédilection du Libre. On n'y voit pas là les besoins du marché grand public, même si ce dernier va être amené à côtoyer, sans le savoir nécessairement, et via des prestataires de services sur Internet, des solutions à base de Libre.

IBM suit ces évolutions de près via notre veille technologique. Quand des applications permettant de faire tourner des applicatifs du domaine propriétaire sur des plate-formes libres tourneront[4], nous verrons.

[2] http://www-1.ibm.com/partnerworld/pwhome.nsf/mktgsale/linux_workload.html.
[3] Les autres classes, iSeries et pSeries ne sont pas en reste.
[4] Lindows, par exemple (note du rédacteur).

C'est le marché qui décide. Par exemple, si cela a un sens, WebSphere sera proposé avec d'autres briques logicielles libres comme il est proposé actuellement avec Apache.

RNTL : *IBM s'engage résolument dans le Libre, et dans le même temps annonce un record en terme de brevets déposés.*

MARC JOLY : Oui. Près de 3000 brevets. Les deux ne sont pas incompatibles. Les deux permettent de protéger et de diffuser l'innovation.

3.3 Produits et services offerts.

RNTL : *Vous avez évoqué les serveurs zSeries. Comment se présente concrètement l'offre Libre d'IBM ?*

MARC JOLY : Elle concerne les serveurs, le matériel, le logiciel et les services et support[5].

Serveurs Les classes xSeries, pSeries et iSeries[6] permettent toutes de tourner sous environnement Linux, mais c'est avec la classe zSeries que l'on peut aller le plus en terme de qualité de service, flexibilité et disponibilité comme j'ai pu le décrire plus tôt. Noter qu'IBM ne touche pas au noyau Linux. Dans tous les cas, ce sont des distributions standard (RedHat, SuSE, Caldera, Mandrake, TurboLinux) qui sont installées.

Matériel En ce qui concerne NUMA-Q, les clients qui utilisent *Linux Application Environment* au-dessus du noyau ptx peuvent faire tourner aussi bien des applications Linux que non-Linux. Pour AIX 5L, il y a compatibilité des sources, conformité avec les standards émergeants, ainsi qu'un environnement de développement qui permet, justement, le développement et le déploiement de solutions Linux sur AIX.
Les THINKPAD ne sont pas en reste, certains d'entre eux étant fournis avec une distribution Caldera (partenaire d'IBM, en l'occurrence) pour offrir des solutions Linux orientées e-business. Dans le même ordre d'idée, IBM NETVISTA THIN CLIENTS sont fournis avec TurboLinux.
Il faut considérer également nos solutions de stockage ENTERPRISE STORAGE SERVER

Logiciel IBM DB2 UNIVERSALE DATABASE dispose de connecteurs pour Linux. Cela permet l'interrogation des bases DB2 sur des *mainframes* à partir d'applications tournant sous Linux.

LOTUS DOMINO tourne également sous Linux, offrant des applications de messagerie et de *workflow*. Toute une série d'applicatifs de la gamme WEBSPHERE apportent au monde Linux de nouvelles solutions d'e-commerce orientées Java. Les MQseries, notamment, développées pour accélérer l'e-business[7].

Les PME ne sont pas oubliées avec l'offre SMALL BUSINESS SUITE FOR LINUX[8] ; ni les développeurs avec DEVELOPER KIT FOR LINUX permettant le développement d'applications java pour les xSeries et les zSeries.

[5] Pour suivre l'actualité de cette offre, http://ibm.com/linux/ est proposé.

[6] Sur les iSeries, Linux complète OS/400, et ne le remplace pas. OS/400 est indispensable pour y faire fonctionner une partition Linux.

[7] http://ibm.com/software/ts/mqseries.

[8] http://ibm.com/linux/sbs.

Services. Dernier volet de l'offre IBM Linux, les services et supports. *IBM Global Services* a développé une offre de services et une expertise disponibles 24h/24, 7j/7, autour de Linux et intégrée à notre interface support classique. Un large spectre de formations, adaptés aux connaissances et compétences de nos clients (des novices aux plus expérimentés) sont offertes. Et nos consultants sont disponibles pour développer des plans directeurs de passage au Libre et déployer des solutions, qui permettent à nos clients de passer de la décision à la production.

RNTL : *Quelle est, justement, l'implication de IBM Global Services ?*

MARC JOLY : IBM Global Services dispose de plus de 300 consultants aux compétences Linux répartis dans le monde. C'est un engagement très fort, qui couvre toutes les étapes d'un passage au Libre, du conseil à la formation, de la stratégie aux tactiques, du déploiement au support. Notre *Linux Technology Center* maintient par ailleurs les pages Web "Linux at IBM", sur http://ibm.com/linux, avec des comparatifs, des études de cas, des nouvelles, des espaces de télé-chargement et des espaces de support électronique. On peut voir ce site Web comme une contribution d'IBM à la communauté Linux.

3.4 La communauté.

RNTL : *Quels rapports peuvent avoir la communauté Logiciel Libre et une grande entité comme IBM ?*

MARC JOLY : Le communauté a longtemps demandé à IBM de s'impliquer dans le Libre, comme elle l'a demandé, d'ailleurs, à d'autres grandes entreprises du secteur. Nous souhaitons que Linux et nos offres classiques soient complémentaires, et nous souhaitons réussir cette complémentarité. Nous ne pouvons pas faire autrement que d'être intégré à cette communauté, d'en accepter les règles, ce qui conforte notre crédibilité. Notre implication dans le projet Eclipse[9] en témoigne, ou encore l'abandon de nos "droits" sur le "FileSystem Management"[10].

Plus de 200 personnes dans notre *Linux Technology Center* travaillent de concert avec la communauté du Libre.

Et puis, il faut bien souligner que sans doute plus de 80 % de ces gens qu'on appelle "communauté du Libre" travaillent en réalité dans des entreprises, des universités. Nous les côtoyons également en tant que clients. Nous n'avons donc pas nécessairement la même vue sur cette "communauté" que celle que pourrait avoir une SSII issue, justement, de cette communauté.

3.5 Modèle économique.

RNTL : *Il ne semble pas y avoir de modèle économique d'IBM particulier en ce qui concerne le Libre : n'est-ce pas tout simplement une application du modèle économique du Libre en vase clos, au sein de IBM Global Services ?*

[9] http://www.eclipse.org/, annoncé fin 2001, dont l'objectif est de développer et promouvoir une plate-forme de développement commune *open-source* ouverte et d'y rallier le plus possible de développeurs et d'éditeurs. Le consortium réunit Borland, RedHat, Fujitsu, Serena, Sybase, Merant, SuSE, QNX, TogetherSoft, Rational, WebGain et bien sûr IBM.

[10] IBM s'implique également dans les initiatives de "Journaled File System".

MARC JOLY : Exactement. Il n'y a pas de modèle économique spécifique. Il y a le modèle économique global d'IBM, et la stratégie Linux décrite plus haut qui s'inscrit dedans.

Il est clair que l'engouement suscité par Linux dans les entreprises est durable. IDC dit que 27 % des nouvelles installations de serveurs en 2000 se sont faites sous Linux, ce qui le place en deuxième position. Nous voyons un chiffre de 40 % d'ici 2005. On peut voir que les éditeurs de logiciels suivent ce mouvement et ont bien compris l'intérêt de leurs clients : nous en voyons près de 2000 (éditeurs) qui ont porté leurs solutions sous Linux. Linux n'est d'ores et déjà plus un marché de niche réservés aux chercheurs, aux étudiants, aux enthousiastes de la première heure. Tous les secteurs, des PME aux grands comptes, s'y intéressent. Et nous voyons très bien cette diversité à travers les projets de nos clients, de la Comtesse du Barry à Opel, de la Compagnie Générale de Géophysique à Aléos.

Comme je l'ai dit plus tôt, Linux et IBM est un mouvement naturel. Que nous avons envisagé il y a plus de deux ans déjà, et qui apparaît totalement intégré à notre offre aujourd'hui.

RNTL : *Les marchés sont-ils faciles à convaincre ? L'avenir du Libre est-il bon en France ?*

MARC JOLY : Oui, il est très prometteur. L'engouement des entreprises, mais aussi des administrations, est croissant. Certains clients sont encore à convaincre, mais peu résistent devant les promesses réelles de réduction du coût de possession de plus de 50 % et d'amélioration du service.

L'avenir du Libre est prometteur, mais il y aura sans doute des concentrations. Par exemple, il y a plus de 200 distributions dans le monde. À terme, seulement une cinquantaine devrait pouvoir couvrir tous les besoins.

RNTL : *Quel chiffre d'affaires IBM fait avec ses solutions Linux ?*

MARC JOLY : Eh bien, ayant investi plus d'un milliard de dollars en 2001, vous avez une idée des ordres de grandeur attendus.

RNTL : *Un an après cet investissement, quel bilan en tirez-vous ?*

MARC JOLY : Cet investissement nous a permis d'intégrer en peu de temps Linux à l'ensemble de notre offre, et d'offrir dans le même temps les services associés pour accompagner nos clients dans le Libre.

- Une partie de cet investissement a été consacrée à l'offre proprement dite.
- Une autre à l'intégration de Linux à nos produits : les classes de serveurs iSeries, pSeries, xSeries et zSeries, DB2, Domino et WebSphere.
- La dernière partie est allée à l'offre de support et de formation, ainsi qu'au *Linux Technology Center* qui dédie près de 200 personnes à Linux.

Nous pouvons dire que nos objectifs ont été atteints.

RNTL : *La stratégie Linux passe-t-elle par des acquisitions ou des prises de participation ?*

MARC JOLY : Pas à ma connaissance pour l'instant.

3.6 La nécessaire promotion.

RNTL : *Comment est assurée la promotion du Libre par IBM ?*

MARC JOLY : Il y a finalement assez peu d'efforts à faire. Les entreprises sont désireuses de faire des économies, et le logiciel libre répond à ce besoin. La complémentarité entre Linux et nos infrastructures matérielles est naturelle. Le prestige d'IBM attire l'attention de la communauté des développeurs. Et l'on peut être assez fier des messages véhiculés par nos publicités, notamment télévisuelles.

La "cellule Linux" est concrètement une division séparée au sein d'IBM. Cela correspond à une centaine de personnes en Europe. Les activités de ces personnes sont réparties entre les axes serveurs, matériels, logiciels, services.

RNTL : *Vous êtes en charge de l'Europe de l'Ouest.*

MARC JOLY : La France, la Belgique et le Luxembourg. J'ai mon équivalent en Allemagne, et dans les pays scandinaves. Avec une stratégie aussi importante que Linux, il est nécessaire d'avoir des gens qui implémentent cette stratégie horizontalement pour l'ensemble d'IBM, serveurs, logiciels et services. C'est mon rôle. Un rôle qui m'amène bien sûr aussi à promouvoir notre stratégie à l'extérieur d'IBM, à travers des conférences, des salons, des discussions en ligne également...

3.7 Une histoire d'hommes et de femmes.

RNTL : *Avez-vous embauché à l'occasion de la mise en place de la stratégie Linux ?*

MARC JOLY : Oui, principalement dans la partie services de notre offre. L'embauche s'est faite aussi bien dans la communauté de développeurs libres qu'ailleurs. Nous avons des gourous techniques, comme on dit. Nous pratiquons une politique de mécénat : des personnes payées par IBM qui travaillent pour des projets communautaires.

RNTL : *Comment cela s'est-il passé avec la culture d'entreprise IBM ?*

MARC JOLY : Nous avons surtout des gens issus des grands systèmes. la culture n'était pas nécessairement la même. Notre phase d'observation de Linux et des logiciels libres[11] a coïncidé en interne avec une phase de sensibilisation, et très vite nos ingénieurs ont été séduits par les modes de développement du Libre et les produits déjà disponibles. Ils sont maintenant tous convaincus des bien-fondés du Libre.

C'est finalement une richesse extraordinaire. On peut sortir des versions avant que tout soit fini. Comme nous faisons les choses passionnément à IBM, avec Linux nous sommes servis.

[11] Une équipe de veille du marché au sein d'IBM a travaillé en amont avant qu'IBM construise son offre Linux.

3.8 IBM et les autres.

RNTL : *Quelles sont vos relations avec les autres acteurs du Libre ? IBM Global Services n'est-il pas un concurrent féroce des SSLL comme on les appelle parfois ?*

MARC JOLY : Comme j'ai pu le dire plus haut, nous travaillons avec plusieurs distributions Linux, et pas directement sur le noyau Linux. Nous n'avons donc pas vocation à développer notre propre distribution. Nous avons des relations avec MandrakeSoft autour des xSeries notamment.

Bull est impliqué dans Linux et est clairement un concurrent pour nous. Ce qui est positif, car plus il y aura de monde sur le marché Linux et plus il se développera.

IBM Global Services ne se pose pas en concurrent. En fait, nous sommes amené à travailler avec les SSII Linux françaises le plus souvent[12].

IBM Global Services est hébergeur sur *mainframe* orientées Linux, mais WorldCom également propose son offre.

IBM a également assisté la jeune pousse Aleos dans le domaine de la fourniture d'applications hébergées (un système S/390 (zSeries), un système de stockage et un système de sauvegarde, là où une autre solution aurait nécessité 100 ou 200 machines). Et ce n'est pas IBM, mais Overlap, un des partenaires *mainframes* en France, qui s'est chargé de la mise en œuvre de la plate-forme retenue. Et, dans le même esprit, Aleos fait appel à des sociétés spécialisées du Lib, comme Alcôve, Aurora, IdealX ou Linagora.

IBM aide également les éditeurs de logiciels, avec un centre de portage présent en région parisienne.

Nous ne pouvons pas aller contre les SSII Linux. Elles sont proches de la communauté et nous souhaitons en être proche également. Elles ont une crédibilité et ensemble nous renforçons la crédibilité des solutions Linux. Elles ont une expertise et nous avons la nôtre. Ensemble, nous sommes des partenaires naturels et gagnants.

RNTL : *Il convient d'ajouter que Marc Joly, interrogé sur la licence IBM (http:// www.opensource.org/ licenses/ ibmpl.html) ne s'est pas étendu, et a porté l'accent plutôt sur les garanties (de 3 ans) offertes sur les z900 par exemple.*

[12] Aussi bien avant le déploiement de la stratégie Linux des testeurs ont pu voir en avant-première le solutions envisagées (cf. chapeau introductif), aussi bien ces tests sont à présent menés à la fois en interne et par nos partenaires.

Les éditeurs.

AYMERIC POULAIN-MAUBANT,
Jipo Interactive, Aymeric.PoulainMaubant@jipo.com.

ACT Europe.

Résumé.

ACT Europe est un acteur-clé du secteur Ada 95, et offre à la fois des produits et des services. Reconnue comme telle, elle est présente sur plus de plates-formes que ses concurrentes et apporte un support utilisateur inégalé. Elle agit ainsi auprès de ses clients qui développent des applications critiques dans des secteurs aussi variés que l'aérospatial, la défense, l'énergie, les transports, la banque, les média et la communication.

Son produit phare est GNAT Pro, environnement de développement pour Ada 95. Cet environnement qui a été développé à partir de la technologie —libre— GNU/gcc, est disponible pour des applications natives ou embarquées, et est fournie avec tout un ensemble d'outils périphériques, de bibliothèques et de documentation.

Mais cet outil ne suffit pas à décrire l'activité d'ACT Europe. En effet, accompagnant le produit se trouve une offre de service, proposant aux utilisateurs des ingénieurs ACT Europe qui seront partenaires et joueront un rôle central dans le projet du client. Un travail commun entre l'équipe client et ACT Europe permet de paramétrer les outils Ada de manière optimale, et d'apporter tout au long du projet l'aide nécessaire sur tous les aspects de développement de logiciels. Cette coopération a des effets immédiats, bénéfiques à la fois pour le client concerné et pour d'autres clients : optimisation et organisation du code, évolution du langage, systèmes multi-langages, mais également réduction des risques, meilleure productivité, *time to market* réduit.

Corollaire : tous les produits livrés sont fournis avec les sources, ce qui n'empêche pas de développer des logiciels propriétaires ou fermés d'une quelconque manière. Deuxième corollaire : ACT Europeparticipe activement à l'effort de la communauté Libre, avec des contributions notables dans le projet GNU d'une part et l'environnement GNOME d'autre part.

– Personne interviewée : Franco Gasperoni
– et site de référence : http://www.act-europe.com/ http://libre.act-europe.fr/

4.1 Quelques rappels.

RNTL : *Qu'est-ce que ACT Europe ?*

FRANCO GASPERONI : ACT Europe est une société unique d'un type un peu nouveau qui a été créée en 1996 conjointement par *Ada Core Technologies Inc.* et les membres européens du projet GNAT Ada 95. Elle fournit un support haut de gamme en Europe autour de l'environnement de développement GNAT Pro Ada 95 pour des clients industriels classiques ou militaires.

Son objectif est bien de fournir à la communauté Ada européenne[1] le plus haut niveau d'expertise possible sur les logiciels Ada et, partant, le meilleur support technique.

Ada Core Technologies elle-même a été fondée en 1994, par les auteurs principaux du compilateur GNAT Ada 95, mais l'expertise remonte aux débuts d'Ada en 1979, concrétisés en 1983 par le premier système Ada validé. Ces travaux préliminaires étaient menés à *New York University*, au sein d'une équipe qui a entièrement été versée dans Ada Core Technologies pour continuer à assurer le succès du langage Ada et s'assurer que les technologies de pointe en matière d'Ada sont toujours proposées aux utilisateurs.

RNTL : *On le voit, Ada 95 est au cœur de votre activité. Quelles en sont ses caractéristiques ?*

[1] Il ne faut pas interpréter ce terme comme pour «Communauté du Libre», mais s'il est employé et gardé ici, c'est bien parce que les racines universitaires d'ACT Europe, et d'Ada en général, entretiennent un rapport particulier avec les utilisateurs d'Ada, même quand ces derniers pourraient être vus par un observateur extérieur et non connaisseur comme typiques du monde logiciel fermé et propriétaire.

FRANCO GASPERONI : Ada 95 est un langage de programmation orienté objet, standardisé ISO/ANSI, et compatible avec son prédécesseur Ada 83. Ce dernier avait été développé à l'origine pour des applications hautement critiques et d'envergure, notamment des applications embarquées. Ses qualités font qu'il a été adopté par des constructeurs de matériel dont la liste donne une idée de ce qu'on veut attendre d'un tel langage : Boeing (777) et Airbus (A340), le TGV, la ligne de métro entièrement automatisée Madeleine-BNF à Paris, ou encore des systèmes embarqués sur satellites.

Ada 95 est une extension de la première mouture Ada 83[2] qui offre une combinaison unique de caractéristiques orientées-objet, temps réel, et programmation concurrente et distribuée. Un tel langage est idéal pour des projets logiciels complexes (plusieurs millions de lignes de code), menés sur le long terme, qui doivent être hautement maintenables, avoir un code réutilisable et extensible, et qui concernent des missions critiques pour lesquels les systèmes informatiques doivent tout simplement être totalement fiables.

RNTL : *Vous avez également évoqué plusieurs fois le terme GNAT.*

FRANCO GASPERONI : GNAT est le système de développement de logiciels Ada (95) le plus répandu, réunissant les forces d'Ada et la «philosophie ouverte» de gcc[3]. En plus des qualités qu'on peut attendre d'un compilateur, il est développé et promu pour attirer de nouveaux utilisateurs Ada et faire en sorte qu'Ada soit associé à tout projet nécessitant un code efficace et sûr. Notre produit GNAT Pro est disponible sur un large spectre de plate-formes, des stations de travail aux PC en passant par des cartes de circuits intégrées vierges[4], et pour des applications conventionnelles ou avancées —temps réel, embarquées, distribuées—, larges (millions de ligne de code) ou modestes (dizaines de milliers).

GNAT repose sur l'environnement —libre— GNU. Ainsi, le compilateur lui-même est intégré dans GCC. Le débuggueur repose sur GDB. L'environnement de développement GLIDE repose sur Emacs. GTKAda GUI repose sur l'environnement GTK et offre des fonctionnalités avancées de construction d'interfaces graphiques. En fait, chacun de ces produits (et ce ne sont là que quelques exemples) reposent sur une «offre» antérieure GNU libre, et propose des fonctionnalités avancées, et adaptées à Ada et aux besoins de ses utilisateurs.

4.2 Stratégie.

RNTL : *Vous venez de brosser en quelques termes-clefs d'où vous veniez et ce que vous faisiez. Comment décririez-vous maintenant votre stratégie ?*

FRANCO GASPERONI : Pour commencer, au-delà de l'objectif général qui est de fournir les produits et les services Ada les plus avancés, nous avons établi un certain nombre de partenariats aussi bien avec des vendeurs de matériels que de logiciels. Pour le matériel, citons par exemple les relations d'ACT et de Silicon Graphics Inc. dont les produits SGI Ada 95 sont fondés sur GNAT, ou encore des relations avec Hewlett

[2]Cette dernière était caractérisée par une idée forte : séparer clairement l'interface (la définition de fonctionnalités), et l'implémentation effective.

[3]GCC est un compilateur, logiciel qui transforme un code-source en un code exécutable par le processeur cible. C'est également en soi un logiciel libre. Il produit un code exécutable de grande qualité, pour de nombreux processeurs cibles, ce qui assure aux langages qu'il supporte (C, C++, Ada, Java notamment) une grande portabilité. Il permet également la compilation croisée, c'est-à-dire qu'il compile du code pour une plate-forme différente de celle sur laquelle il tourne, ce qui permet une grande flexibilité pour les développeurs.

[4]*bare boards*.

Packard et Compaq. Côté logiciel, on citera l'Aérospatiale, British Aerospace, Construccionnes Aeronauticas in Spain, Lockheed, BNP, Canal+, Ericsson, Nortel, Philips Semiconductor, l'US Air Force Academy et West Point, ce qui donne une idée des secteurs couverts.

RNTL : *Vous dites «partenariats». L'idée est d'établir une relation à long terme avec vos clients...*

FRANCO GASPERONI : Exactement. Il ne s'agit pas de vendre un produit à un client et de le laisser seul ensuite avec. Le support est le cœur de notre activité.

Dans le mode propriétaire, tout l'effort est porté sur la vente du produit, entraînant des coûts marketing très importants. Dans ce mode, l'offre de support est en fait une charge pour le vendeur, alors que dans le même temps, utilisateurs et développeurs ont de plus en plus besoin d'aide pour utiliser au mieux les produits qu'ils achètent.

Nous pensons que, dès lors qu'on recherche le meilleure équation entre la qualité et le *time-to-market*, le support est la clé. La technologie actuelle apporte un niveau de complexité et de besoin d'information inégalé, et le support est ce qui permet d'aborder sereinement cette complexité. C'est pourquoi nous en faisons le cœur de notre activité. Un support pour des projets-clients qui ont des durées de vie de plusieurs dizaines d'années...

RNTL : *Comment implémentez-vous ce support au quotidien ?*

FRANCO GASPERONI : On pourra sans doute le détailler plus tard, mais en quelques mots, cela va des questions sur l'utilisation d'Ada, à des explications sur comment utiliser efficacement nos outils, d'analyses de rapports de problèmes à des développements et améliorations spécifiques, de rendez-vous réguliers à une assistance 7j/7.

RNTL : *Qu'apporte le mode logiciel libre ?*

FRANCO GASPERONI : Toute la différence.

Nous pensons que le monde Ada tire parti d'une technologie non-propriétaire. Et nous livrons effectivement tous nos produits avec les sources.

Nos clients doivent fournir des applications logicielles fiables, qui doivent rester fiables de nombreuses années. Ils ne peuvent pas se reposer sur des offres logicielles de type boîte noire et des éditeurs qui peuvent modifier les caractéristiques de leurs produits dans un sens non souhaitable. De plus, ils ne sont pas censés être des experts sur l'ensemble du domaine d'un langage, et en particulier son cœur ; ils doivent en effet se concentrer sur le cahier des charges des logiciels qu'ils développent et sur le métier de leur propre client (eux-mêmes, peut-être).

Le fonctionnement ouvert que nous privilégions leur offre l'assurance d'avoir un contrôle permanent de tous les composants de leur application finale, et de pouvoir accéder à tout instant aux développeurs qui ont construits ces composants.

Un autre bénéfice immédiat, qui lui aussi constitue une part importante de notre stratégie, est de permettre l'émergence de *pool* locaux d'expertise. Nous pensons avoir les meilleurs chez nous, mais il existe une concurrence, des gens compétents qui connaissent bien GNAT également, certains d'entre eux travaillant chez ou avec nos clients.

Enfin, et c'est une assurance à la fois pour nous et nos clients, nos codes ouverts sont relus, examinés avec

soin par une large communauté d'experts Ada, notamment en université. Nous travaillons dur également pour que le jugement de ces pairs soit toujours positif, et qu'ils apprécient ce que nous produisons, aussi bien ce que fait le logiciel, que la manière dont il est écrit.

4.3 Produits et services offerts.

RNTL : *Venons-en à votre offre, que nous allons passer en revue. Comment se présente-t-elle ?*

FRANCO GASPERONI : Concernant les produits, nous offrons :
- *GNAT Pro*, environnement complet de développement Ada 95, contenant notamment la chaîne de compilation, l'éditeur avancé Glide, et le débuggueur
- *GNAT Pro High-Integrity Edition* spécialisée pour les systèmes hautement critiques
- *GNAT Pro Bare Board*
- *GNAT Pro for Open VMS*
- *GNAT Pro for VxWorks*

auxquels on ajoutera :
- CGNAT, environnement de développement GNU C
- GtkAda, permettant de développer des interfaces graphiques modernes pour X-Window et Windows
- JGNAT, Ada pour plateformes Java
- GLADE, GNAT en systèmes distribués
- GNATCOM, permettant de développer des composants CON+/DCOM/ActiveX adsn des applications Ada
- ASIS-for-GNAT, pour développer des sur-outils[5] Ada
- XML/Ada, pour traiter des flux XML dans les applications Ada

RNTL : *Tous ces produits étant fournis avec leur support.*

FRANCO GASPERONI : Tous ces produits étant fournis avec leur support. Ce qui se décline de la manière suivante :
- GNAT Partner, lien entre l'équipe client et les ingénieurs ACT Europe
- GNAT Enterprise, pour une personnalisation fine sur un projet, ou bien des projets au calendrier serré.
- GNAT 24/7, dont le nom laisse deviner l'étendue du support offert
- GNAT Assurance : le code produit par le client est intégré dans la démarche qualité d'ACT Europe.

Le support consiste aussi en du conseil classique et de la formation sur site.

RNTL : *Vous disiez qu'ACT Europe était unique en son genre.*

FRANCO GASPERONI : Oui, car comme vous le voyez un rôle central y est donné au support sur les projets de nos clients, auxquels participent nos ingénieurs, les équipes de développement étant ainsi mixtes.

Ce travail «au plus près» est l'assurance que le client va utiliser nos produits de manière optimale, et qu'il sera assisté dans toutes les phases et pour tous les aspects de son développement. En plus de cet échange concernant le projet du client, nos équipes s'occupent également de l'optimisation et l'organisation du code,

[5] Permettant par exemple de créer des outils de test ou de mesure de logiciels Ada.

de l'évolution du langage, des aspects liés aux systèmes multi-langages, et ceci dans une logique de réseaux, les avancées ayant été faites chez l'un pouvant bénéficier à l'ensemble des clients. Ces principes apportent une réduction des risques, une meilleure productivité, et un *time to market* réduit.

Typiquement, tous nos produits sont fournis avec un support projet qui est un contrat annuel contenant :

- Le CD-ROM du (des) logiciel(s) pour la plate-forme cible, ainsi que la documentation papier,
- Un accès FTP valable un an pour mettre à jour le produit et sa documentation,
- Une année d'abonnement à deux listes de groupes d'utilisateurs, ug@gnat.com et ug@act-europe.fr,
- Un an de "GNAT Product Support" et un an de "GNAT Project Support"

RNTL : *Et que trouve-t-on dans ces deux offres —Product et Project— de support ?*

FRANCO GASPERONI : Pour Product, une aide pour les procédures d'installation, des explications concernant les fonctionnalités dépendantes de l'implémentation choisie, la correction des problèmes éventuellement rencontrés lors de l'utilisation du produit, un extranet pour y déposer des rapports de problèmes et accéder à des pré-versions de produits.

Pour Project, une aide plus soutenue sur tous les aspects concernant le développement en Ada, y compris les différences entre Ada 83 et 95, des conseils sur comment utiliser au mieux Ada 95, GNAT Pro et les outils qui l'accompagnent, comment les configurer, comment migrer d'une technologie non Ada vers Ada, comment lier Ada et C ou C++.

De même, tous produits «add-on» comment CGNAT, GtkAda... ont une offre de support de ce type adaptée à leurs caractéristiques.

RNTL : *Ce sont là des offres de support classiques, si l'on peut dire...*

FRANCO GASPERONI : ...Auxquels nous avons ajouté les supports Partner, Enterprise, 24/7 et Assurance, que je vous indiquais plus tôt.

RNTL : *Alors, quelles sont les caractéristiques qui font leur différence avec du support plus généraliste ?*

FRANCO GASPERONI : D'abord, tous ces contrats sont annuels et rattachés à un seul projet à la fois.

Avec GNAT Partner, un technicien senior d'ACT représente le projet du client au sein de notre équipe, quelqu'un avec qui le client peut parler directement, qui analyse avec lui les différents problèmes qu'il rencontre, et qui le tient au courant des derniers développements de la technologie GNAT. Ce support est accompagné de deux jours de formation sur site, plus des rendez-vous téléphoniques réguliers.

GNAT Enterprise comprend l'offre GNAT Partner, avec trois jours de plus de formation et de conseil. C'est surtout une offre de support pour des projets aux *deadlines* serrées, ou qui nécessitent une personnalisation voire même des fonctionnalités étendues de GNAT. 3 personnes/semaine sont ainsi incluses dans ce support pour apporter les améliorations souhaitées, ou venir chez le client résoudre des points bloquants que l'équipe support à distance n'aurait su traiter.

GNAT 24/7 étend l'offre GNAT Entreprise avec un support continu par téléphone, et un temps de réponse garanti de 90 minutes.

Enfin, GNAT Assurance est une offre de support spéciale qui place les applications de nos clients dans notre propre processus interne d'assurance qualité. Ainsi, les produits de nos clients ayant souscrit un tel support sont inclus dans nos tests de régression, de manière à ce que les futures versions de GNAT Pro n'affectent ni le comportement ni les performances de ces produits. Par ailleurs, dès que le client rencontre un problème, il n'a pas à isoler le problème lui-même : il nous signale ce qu'il observe, et c'est à nous d'isoler précisément le problème et la section de code concernée.

RNTL : *En plus de cette large offre de support, vous avez une offre de services.*

FRANCO GASPERONI : Comme du portage sur des environnements spécifiques, de la personnalisation de nos outils ou l'ajout de fonctionnalités. Mais également du conseil et de la formation (entre un et cinq jours), sur et hors-site.

4.4 Modèle économique.

RNTL : *Parlons à présent de votre modèle économique. Pour commencer, vous proveniez de l'université.*

FRANCO GASPERONI : Effectivement, dès 1994 aux États-Unis, et 96 en Europe, en réponse à une demande croissante d'un support commercial autour de GNAT. Nous avons commencé avec quelques prêts minimes classiques et existons depuis le début avec comme seul financement les revenus de notre activité. Des revenus qui sont plus importants que ce que nous avions imaginé à l'origine, et qui font d'ACT une société qui tourne bien.

Plus précisément, la plus grande partie des revenus provenant de nos offres de support, nous nous devons d'une part d'assurer le meilleur support possible à nos clients, et d'autre part d'avoir un taux de renouvellement d'au moins 90 %, les 10 % restant correspondant à des projets qui sont entièrement terminés.

Ce modèle fait que nous avons d'année en année un flux de revenus stable et prédictible, et que cela peut durer indéfiniment[6].

Par ailleurs, plus de 90 % des personnels sont des techniciens de haut niveau, le reste composant une petite équipe marketing et commerciale efficace. L'ensemble donne une entreprise ayant des frais généraux réduits. Nous pouvons donc nous contenter d'un flux de revenus raisonnable tout en maintenant une offre de grande qualité.

RNTL : *Quels prix pratiquez-vous ?*

FRANCO GASPERONI : Je dirai «les prix n'ont rien à voir avec le logiciel libre». L'important ici, ce ne sont pas les prix, mais la liberté offerte avec le code-source. Que vous payiez ou non pour avoir une copie de notre logiciel libre, ce que vous avez de toute façon, c'est la liberté de le copier, le modifier et de le redistribuer.

Donc, de manière naturelle, le prix peut être de zéro. Dans le mode propriétaire, vous payez une licence

[6]Tant que la technologie Ada est porteuse, bien sûr, mais il faut rappeler qu'ACT met également tout en œuvre pour améliorer GNAT et le faire évoluer avec les besoins des clients, gardant ainsi ce système toujours à la pointe. Et ceci se fait à travers les offres de support, justement.

d'utilisation d'un logiciel, puis vous payez de la maintenance. C'est le cas pour GNAT, avec une licence d'exploitation de zéro. Ce qui ne veut pas dire que le coût d'utilisation de GNAT est de zéro. Surtout pour de gros projets logiciels qui impliquent nécessairement un support, si possible de la part de ceux qui sont les développeurs originels des logiciels de départ.

J'en profite pour faire un aparté. C'est histoire de licence gratuite a des avantages immédiats pour tout le monde, aussi bien nous-mêmes que nos clients qui vont nous solliciter fortement pour du support. Comme nos logiciels sont disponibles pour tous, on trouve sur le marché une communauté grandissante d'utilisateurs et de développeurs qui maîtrisent de mieux en mieux Ada et GNAT. Et cette communauté, enthousiaste, offre des contributions majeures pour l'amélioration des produits. Également, étant libres d'accès, nos produits peuvent être intensivement testés, sans aucune restriction, avant d'être définitivement adoptés.

Et tout ceci ne détruit pas le besoin d'une offre de support.

RNTL : *Quel est votre chiffre d'affaires ?*

FRANCO GASPERONI : Entre 3 et 5 millions d'Euros, autrement-dit cinq fois moins que si le système était en mode propriétaire.

RNTL : *Des clients vous ont-ils demandé spécifiquement de développer en Libre ?*

FRANCO GASPERONI : Oui, il y en a pour lesquels c'est le critère numéro un, car il leur donne la garantie que le savoir-faire sera disponible à travers de nombreuses personnes, et pas seulement un petit groupe d'une dizaine d'experts.

RNTL : *Un tel modèle économique peut-il tenir quand les concurrents peuvent facilement reprendre les codes-sources de vos logiciels ?*

FRANCO GASPERONI : Pour l'instant, c'est le cas.

Les concurrents peuvent d'ailleurs le faire de manière illégale (au sens de la GPL, c'est-à-dire en rendant propriétaire l'ensemble de leur production y compris ce qui provenait du libre), mais à la limite ce n'est pas grave, car avoir une concurrence, ne pas être le seul sur le marché, est toujours préférable.

Et puis, il faut accepter de ne plus raisonner systématiquement en parts de marché. Nous ne sommes plus dans cette —seule— logique. Dans notre environnement, c'est plus une logique de don / contre-don qu'il faut adopter que la logique économique classique. Ce n'est donc pas tant la récupération de «parts de marchés» par des concurrents ayant repris nos codes qui nous intéresse que celle des «parts de confiance» que nous possédons dans l'ensemble de la communauté des utilisateurs. C'est un autre écosystème, un autre équilibre. Dans lequel les créateurs initiaux[7] ont un avantage pendant longtemps au début, et peuvent le conserver tant qu'ils s'engagent fortement dans l'innovation et dans l'accompagnement au plus près de leurs clients.

En ce sens, ACT est et reste le leader technique pour tout ce qui est développement en Ada 95. Certes, j'ai beaucoup parlé de notre offre de support, mais cela ne nous empêche pas d'investir de manière non négligeable dans le développement en continu, ce qui nous place très en avant de nos concurrents. Nous sommes ainsi les seuls à offrir un support sur l'ensemble d'Ada 95 y compris ses annexes (*information systems, safety and security, distribution*), et des versions pour VMS, Linux, OS/2 et Solaris x86.

C'est sûr, le marché que nous visons nous aide. Une telle chose serait plus difficile sur le marché grand

[7] Noter que les 5 fondateurs initiaux sont toujours présents à ACT Europe.

public. Ici, nous pouvons rester des ingénieurs, honnêtes et engagés jusqu'au bout, sans avoir à se lancer dans des chemins marketing tortueux.

Donc, oui, pour nous un tel système tient.

4.5 Une histoire d'hommes et de femmes.

RNTL : *Pour finir, quelques mots sur les aspects humains. Quels sont vos rapports avec la communauté ?*

FRANCO GASPERONI : Nous avons un engagement fort vers la communauté[8], qu'on retrouve décliné sur notre site Web communautaire http://libre.act-europe.fr/. Nous y proposons des outils aussi bien pour les développeurs que pour les professeurs et les étudiants. C'est en particulier pour cette deuxième catégorie qu'un site communautaire, aisément accessible, est important. C'est à partir de ce site que l'on peut télécharger nos produits en version complète, mais sans support bien sûr.

RNTL : *Et au sein d'ACT Europe, comment sont gérées les personnes ?*

FRANCO GASPERONI : C'est une organisation qui reflète bien les principes du Libre, tout simplement. Nous avons une structure plate, sans hiérarchie. Nous travaillons avec des gens mûrs et responsables, qui ont beaucoup d'indépendance. Nous avons des employés qui ont eu des idées intéressantes, les ont développées de leur côté, puis ACT Europe s'y est intéressée en tant que telle, et l'ensemble est maintenant une offre ACT Europe . C'est le cas par exemple avec toute l'offre autour d'XML.

4.6 En guise de conclusion.

RNTL : *Un dernier mot ?*

FRANCO GASPERONI : ACT Europe est une entreprise qui a trois vocations. Elle se consacre d'une part à ses clients, qui ont des projets complexes, sur des durées de vie de 5, 10 parfois 20 ans. Elle se consacre à Ada, avec comme volonté de rendre Ada accessible. Et elle se consacre au logiciel libre, qu'il s'exécute d'ailleurs sur des systèmes libres ou non.

[8] NDLR : Il faut noter que, de temps à autres, ce qu'on pourrait appeler de petits noyaux communautaires apparaissent de façon spontanée, en particulier au sein de la communauté scientifi que. On peut alors se dire qu'une partie de la communauté est incarnée à tel endroit, plutôt que se la représenter de manière diffuse au sein d'Internet. Ada, et ses origines universitaires, sont probablement la raison de ces phénomènes.

MandrakeSoft.

4.7 MandrakeSoft.

- Personne interviewée : Jacques Le Marois, PDG, co-fondateur
- Sites de référence : http ://www.mandrakesoft.com/

Avertissement : Le site Linux-Mandrake[9] a été sélectionné parmi les 100 meilleurs sites Linux choisis en juin 2000 par Linux Magazine. Il a également obtenu un *Editor's choice* émanant de l'*Open Directory Project*. Il mérite amplement ces récompenses, à la fois parce que ce site (et en fait toute une galaxie de sites, comme on le verra plus loin) est clair et complet en ce qui concerne ce qu'on est en droit d'attendre d'un site commercial, mais il va plus loin, donnant de nombreuses informations sur le fonctionnement de la société (son introduction en bourse est passée par là) et sur ses engagements.

Dans l'esprit libre le plus pure (emprunter, adapter, citer ses sources), nous avons choisi de ne pas réinventer la roue et de reprendre ici de larges pans de ce site dès lors qu'ils étaient pertinents.

4.7.1 Activité.

Fondé en 1998, MandrakeSoft est une société de services informatiques spécialisée sur le système d'exploitation Linux. L'activité s'organise autour de 2 pôles :

- *utilitaires* (97 % du CA). Les utilitaires facilitent l'installation, la configuration et l'administration du noyau du systèmes d'exploitation Linux. On peut décomposer ce pôle comme suit :
  - *ventes retail* au travers d'un réseau de grossistes et de distributeurs (68 % de l'activité actuellement contre 78 % en 2001)
  - *ventes online* sur le site MandrakeStore[10], plus rémunératrices en terme de marge brute (28 % de l'activité actuellement contre 9 % sur 4 mois d'activité en 2001)
  - *ventes OEM* concernant 3 % du chiffre d'affaires
- *services* (3 %) : support Web et téléphonique à accès limité.

Par zone géographique, le CA se ventile comme suit : Amérique du Nord (73 %), Europe (25 %) et autre (2 %).

MandrakeSoft édite le système d'exploitation Mandrake Linux sur le mode du développement ouvert (accès libre au code source publié selon les termes de la General Public License). Plusieurs centaines de développeurs dans le monde contribuent à l'amélioration constante du produit via Internet. Mandrake Linux est la distribution Linux la plus internationalisée : elle est supportée (à l'installation) dans près de 50 langues. Mandrake Linux comporte des interfaces graphiques performantes déjà configurées (notamment les deux leaders, KDE et Gnome) et plus de 2300 applications, dont les célèbres Netscape Communicator, la suite bureautique StarOffice et le serveur Web Apache.

Les collaborateurs de MandrakeSoft sont répartis dans plus de 20 pays, mais la société est elle-même physiquement présente seulement en France, aux États-Unis, au canada et au Royaume-Uni. Depuis le 30 juillet 2001, la société est inscrite au Marché Libre d'Euronext Paris (code Euroclear : 4477.PA MandrakeSoft/Boursorama [2002]).

MandrakeSoft se définit à la fois comme un animateur de projets et un fédérateur de compétences dans le domaine de l'Open Source. Son ambition est de faire bénéficier les utilisateurs des innovations issues des travaux de la communauté, qui réunit plusieurs centaines de personnes à travers le monde, en proposant un système d'exploitation puissant, stable associé à une gamme complète de solutions pour les postes clients (environnement graphique de type Windows, suite bureautique compatible avec tout format de

[9] http://www.linux-mandrake.com/.
[10] http://www.mandrakestore.com/.

fichier, navigateur Internet) et pour les serveurs (serveur Web, serveur de messagerie, serveur de fichiers et d'impression...).

Aujourd'hui, Mandrake Linux est la distribution[11] Linux la plus internationalisée puisqu'elle est supportée dans près de 50 langues à l'installation. De plus, Linux-Mandrake garantit une installation et une «prise en main» simples, rapides et fiables. Les différentes solutions Linux-Mandrake regroupent le système d'exploitation, les programmes associés sur des CD-ROM, les outils d'installation, une documentation complète ainsi que du support. Mais MandrakeSoft va encore plus loin en proposant à ses clients (aussi bien des personnes physiques que des associations, des collectivités et des entreprises) un accompagnement global en termes de services sous forme de solutions packagées et / ou de solutions sur mesure[12].

MandrakeSoft revendique plusieurs millions d'utilisateurs personnels et plusieurs milliers d'entreprises utilisatrices.

4.7.2 Historique.

La création de MandrakeSoft en novembre 1998 résulte de la rencontre via Internet de jeunes passionnés du système d'exploitation Linux : Gaël Duval, Frédéric Bastok et Jacques Le Marois. Gaël Duval avait placé sur Internet quelques mois auparavant une première version de la distribution Mandrake, fondée sur la également célèbre RedHat, et le succès avait été immédiat, des efforts ayant été particulièrement déployés sur le système d'installation. Disponible en téléchargement, la distribution de la toute jeune société est très rapidement et largement disséminée sur la planète, et dès février 99 une nouvelle version *packagée*, avec manuel d'utilisation, est disponible.

Ces premiers succès attirent les partenaires, et notamment un réseau de distribution relativement important (Macmillan Software, nouvellement appelée Pearson Technology Group) aux États-Unis, ce qui lui permet de sortir de France et de s'implanter durablement outre-Atlantique (ouverture en septembre 1999 d'une filiale aux États-Unis).

Toute l'année 1999 est émaillée de récompenses, qui renforcent l'estime que lui portent ses utilisateurs, et ceci ne sera pas démenti en 2000. En deux ans, cette société française s'impose comme la deuxième référence mondiale en matière de logiciels Open Source et Linux. L'année 2000 est celle de la croissance, avec l'ouverture de bureaux et de filiales (Londres, Canada), et les accords de partenariat (revendeurs à valeur ajoutée et centres de formation).

L'année 2001 voit le développement du portefeuille d'offres, ouvert l'année précédente avec les offres de formation et de support technique packagées, et qui se poursuivent avec l'ouverture des plates-formes communautaires en ligne associées, ainsi qu'une plate-forme de commerce électronique. Par ailleurs, des accords OEM[13] sont signés avec Compaq et HP. Enfin, le fait marquant de l'année 2001 est l'introduction en bourse (Marché Libre d'Euronext Paris), que l'on traitera plus longuement plus loin.

[11] Il ne faut pas confondre «distribution» et simple «assemblage». Le produit de Mandrake est plus que la simple réunion de logiciels libres ; un système d'installation et de confi guration développé exprès est proposé et fait le succès du produit. Pour beaucoup d'utilisateurs, cette distribution s'apparente au couteau suisse des distributions, tant elle est modulable.

[12] notamment dans des organismes de l'État comme le ministère de la culture ou l'armée de l'air.

[13] *Original Equipment Manufacturer*.

4.7.3 Produits.

La mission de MandrakeSoft est simple : proposer une offre innovante globale associant des produits exclusivement Open Source et des services adaptés à ses utilisateurs en leur apportant ainsi tous les bénéfices de la puissance, de la performance et de la simplicité d'utilisation du système d'exploitation Mandrake Linux.

– *Marché Grand public* Début 2002, l'offre consiste en :
  – Mandrake Linux Standard édition 8.1, à la fois conçue pour les débutants et les utilisateurs plus expérimentés. Elle offre les outils de base pour la découverte et l'utilisation des solutions Linux
  – Mandrake Linux PowerPack édition 8.1, un *pack* complet avec plus de 2300 applications sur 7 CDs, associé à un support de 30 jours par téléphone et 60 par Internet.
– *Marché Entreprise (PME, Grands Comptes)* Début 2002, l'offre consiste en Mandrake Linux ProSuite édition 8.1, conçue pour les petites et moyennes entreprises, mais facilement adaptable pour le déploiement en grande entreprise. Elle offre un support technique étendu. Elle met l'accent sur les outils nécessaires à la mise en place d'un environnement informatique complet et l'assurance d'une solution stable et sécurisée pour les serveurs et stations de travail
– *Produit OEM* Fondé sur Mandrake Linux Desktop, le produit OEM a été spécialement élaboré afin que les constructeurs informatiques proposent à l'utilisateur final une solution matérielle et logicielle clé en main : pré-installation de Mandrake Linux sur les matériels constructeurs et support technique associé.
– *Offre ISV*[14]. L'offre ISV est une offre aux éditeurs et développeurs de logiciels, qui bénéficient via le navire amiral Mandrake d'une croissance rapide du marché Linux. Il s'agit d'un programme de partenariat qui apporte une plus grande visibilité aux produits de l'ISV à travers, par exemple, le référencement dans le catalogue Mandrake de solutions logicielles (Solutions Kit), jusqu'à une collaboration technique et commerciale complète et un support technique de haut niveau

Il convient de souligner que MandrakeSoftpoursuit depuis la toute première version de Mandrake Linux une politique d'innovations technologiques. Elle ne se contente pas d'assembler des logiciels pour constituer sa distribution, mais elle s'attache à créer les liants technologiques (par exemple les systèmes d'installation) qui en font un outil facile d'usage et ont contribué à sa réputation, d'autant plus que la compatibilité avec la distribution-mère, RedHat, restait de mise.

Les outils suivants ont par exemple bénéficié de l'expertise conjuguée des équipes de Recherche et Développement de MandrakeSoft et des contributeurs internationaux de Linux-Mandrake. L'utilisateur non-averti de Linux appréciera les effort menés par ce département R&D et l'implication de la communauté.

– *DrakX* est un outil qui a pour rôle de faciliter l'installation de la distribution Mandrake Linux depuis un CD-Rom, le réseau ou un disque dur tout en exploitant une interface graphique très conviviale et simple d'emploi.
– *Lnx4Win* gère l'installation de Linux sur une partition Windows (95 ou 98).
– *DiskDrake*, utilitaire de partitionnement de disque dur pour PC qui sert à partager facilement les unités de stockages en plusieurs parties à l'aide de la souris. Les informations des partitions sont ensuite présentées dans une interface visuelle. Cette fonctionnalité est intégrée à DrakX.
– *HardDrake*, projet ambitieux visant à faciliter la reconnaissance, par le système d'exploitation, des périphériques internes du PC ainsi que leur configuration. Là aussi, une interface graphique vient au secours des utilisateurs.
– *RpmDrake* simplifie l'installation ou la suppression de logiciels.
– *DrakConf* gère la «post-configuration» des périphériques.

[14]Independent Software Vendors

4.7.4 L'offre de conseil.

MandrakeSoftne se contente pas d'offrir des produits. Elle leur associe une offre de conseil, dédiée aux entreprises. Elle peut ainsi aider les entreprises à construire leur projet et lui apporter une solution adaptée, issue de son expérience d'éditeur : sécurisation de l'environnement informatique, migration vers Linux, personnalisation de distributions, déploiement de solutions Open Source (serveur Web, mail, partage de fichiers...).

Les interventions de conseil s'effectuent en mode projet, au forfait ou en régie. Elles sont proposées conjointement aux offres de formation et de support, en fonction des besoins identifiés. De plus, le réseau de partenaires de MandrakeSoftest également impliqué et peut ainsi offrir une présence rapprochée.

Cette offre de conseil est supportée en amont[15] et en continu[16] par le site Web de cas d'écoles :

http://www.mandrakebizcases.com/

4.7.5 L'offre de formation.

Dans la même logique, MandrakeSoft offre des formations hors-ligne et en ligne.

Cette formation dispensée dans des centres répond à la fois aux besoins des utilisateurs finaux et des administrateurs systèmes. Elle offre des stages constamment mis à jour sur les solutions Linux. Le programme de la formation Linux-Campus est disponible à travers un réseau de Centres certifiés partenaires.

Linux-Campus est une référence dans l'univers des formations Linux, renforcée par son partenariat avec le Linux Professional Institute (LPI) qui propose une certification professionnelle Linux, mondialement reconnue.

http://www.mandrakecampus.com/

MandrakeCampus vise à fournir à la communauté GNU-Linux, depuis les nouveaux utilisateurs jusqu'aux experts, des formations et des cours sur tout ce qui touche aux logiciels libres. Grâce à cette plate-forme électronique, les enseignants, les tuteurs et les apprenants bénéficient d'un lieu où ils peuvent partager le savoir. Pour conserver l'esprit Open Source qui fait la force de MandrakeSoft et de sa communauté d'utilisateurs, Mandrake et Logidée sont en train de finaliser un accord qui permettra de mettre le contenu de Mandrake-Campus.com sous licence GFDL (GNU Free Documentation License). Il convient de noter l'effort porté sur les tutoriaux en ligne, tout en images et de très grande qualité.

4.7.6 L'offre de support.

Le conseil et la formation peuvent ne pas suffire sur le long terme, et c'est pourquoi MandrakeSoft propose également des prestations de support de plusieurs natures, sans compter les espaces propices aux échanges de savoir entre les utilisateurs.

Aux entreprises, MandrakeSoft propose un ensemble de services professionnels parmi lesquels figure un support de haut niveau assurant l'installation et la configuration rapides d'un environnement professionnel complet. Pour tout le monde, Mandrake Linux propose un autre site Web :

http://www.mandrakeexpert.com/

plate-forme en ligne qui permet aux utilisateurs de poser des questions et d'obtenir des réponses de la part d'experts. On peut tour à tour y jouer le rôle d'expert ou d'utilisateur en quête d'informations.

[15] c'est-à-dire que ce site Web permet aux prospects de trouver des premiers éléments de réponse et de juger très rapidement de la qualité des conseils Mandrake.

[16] s'enrichissant au fur et à mesure de nouveaux témoignages.

Bien sûr, un support gratuit est livré avec chaque pack Mandrake Linux, activable après avoir enregistré le dit pack[17]

4.7.7 Concurrents.

On peut définir deux types de concurrents (deux ensembles à intersection non nulle, d'ailleurs) : les distributions (Debian, Redhat...) et les vendeurs de distributions (et/ou de services associés). Ce deuxième type de concurrence est une concurrence classique entre entreprises, tandis que le premier fait plus intervenir l'affectif. Issu de la communauté et de la distribution RedHat, cet affectif est encore très présent dans les choix effectués par les clients et dans l'opinion que la communauté peut avoir de Mandrake et de ses choix stratégiques (passage en bourse, appel à souscription au club d'utilisateurs, voir infra.). Il conviendrait de faire une histoire des distributions[18] pour commencer à cerner cette donnée affective.

Ceci dit, plusieurs sondages récents montrent que Mandrake Linux a réussi en moins de trois ans à être l'une des versions de Linux la plus utilisée au monde :

Linux Magazine UK	Linux.ie (Ireland) - Juillet 2001[a]	PCMag.com - 8 octobre 2001 - US[b]	Linux.com (US) - 24 septembre 2001[c]	LinuxUser Magazin - Novembre 2001 (Allemagne)
Étude exhaustive faite auprès de 35.000 lecteurs	Quelle distribution préférez-vous ?	Quelle distribution préférez-vous ?	Quelle est la meilleure distribution pour un novice ?	Quelle distribution utilisez-vous régulièrement ?
Mandrake : 46.02 % Red Hat : 21.33 % SuSE : 18.67 % Debian : 5.33 % Corel : 2.66 % Caldera : 2.66 % Autres : 3.33 %	Mandrake : 35 % Debian : 23 % Redhat : 18 % SuSE : 18 % Caldera : 1 % Corel : 1 % Autres : 1 %	Mandrake : 43,78 % Redhat : 27,9 % SuSE : 15,41 % Libranet : 1,42 % Autre : 11,49 %	Mandrake - 36,7 % Redhat : 33,5 % SuSE : 13,5 % Slackware : 5,6 % Debian : 4,6 % Caldera - 3,7 %	SuSE : 76,2 % Mandrake : 33,8 % Redhat : 19,7 % Debian : 10,7 % Slackware : 2,8 % Caldera : 2,4 % EasyLinux : 2,1 % Autres : 2,8 %

[a] http://www.linux.ie/polls/pollresults.php?recount_pollId=1
[b] http://www.pcmag.com/poll_archive/0,3044,p~\%253D1035~\%2526bn~\%253D1,00.asp
[c] http://www.linux.com/polls/index.phtml?pid=128

Ces sondages ont été réalisés auprès des individus et non pas des entreprises. C'est grâce à son travail sur la facilité d'installation et d'utilisation de Linux que MandrakeSoft a réussi à prendre cette position. Cette base

[17] ce qui permet à Mandrake de suivre en partie la dissémination de ses produits, une tâche que l'on sait particulièrement diffi cile avec les logiciels libres et gratuits, et qui a longtemps été une faiblesse par rapport à des éditeurs de systèmes propriétaires pré-installés qui se targuent —eux—d'un nombre d'utilisateurs forcément sur-estimé.
[18] À partir de http://www.linux.org/dist/list.html.

importante d'utilisateurs est un atout pour le développement de MandrakeSoft sur le marché des entreprises. Chaque utilisateur satisfait est un prescripteur.

4.7.8 Explications d'un succès.

Plusieurs facteurs expliquent la croissance exceptionnelle de l'entreprise, passée de 3 à plus de 120 salariés en 2 ans : dans le respect de la philosophie «Open Source», MandrakeSoft met librement à disposition sur le Web le code source de sa distribution Linux-Mandrake. Ce sont ainsi plusieurs centaines de développeurs indépendants dans le monde qui améliorent en permanence le produit et en font une des solutions Linux les plus simples, complètes et puissantes.

Le modèle économique de l'entreprise repose ainsi en partie sur la rapidité de développement à l'échelle planétaire et fait de Linux-Mandrake la distribution la plus internationalisée qui soit compte-tenu de la provenance de ces contributeurs ; ce qui en fait la distribution qui essaime le plus rapidement possible.

De plus la société est implantée en France, aux États-Unis, au Québec et en Angleterre. L'excellence technologique des produits de MandrakeSoft est reconnue par les spécialistes du marché. Linux-Mandrake a été désignée «Meilleur produit de l'année» et «Meilleure distribution pour serveur» lors du salon LinuxWorld Expo 1999 ; elle a reçu en avril 2000 le «Platinum Award» de la revue britannique PC Answers ainsi que l'Editor Choice Award de la célèbre revue américaine Linux Magazine en septembre 2000. Numéro 1 en France, Linux-Mandrake s'est également positionnée en première place des ventes de produits Linux aux États-Unis trois mois de suite en 2000 avec plus de 30 % du marché Nord américain (Source : PC Data). La réussite de MandrakeSoft démontre que l'industrie américaine du logiciel peut facilement être concurrencée à partir de la France. MandrakeSoft a été agréée «entreprise innovante» par l'Anvar. Elle est l'une des premières sociétés éditeurs de distributions Linux à avoir reçu cet agrément pour son savoir-faire en logiciel Open Source. Enfin, MandrakeSoft s'implique fortement dans de nombreuses actions ayant pour but la défense et la promotion du logiciel libre et de l'Open Source.

Ce succès auprès des utilisateurs a attiré de nombreux investisseurs[19], des partenaires[20], et créé des alliances stratégiques et technologiques [21], sans compter des partenariats avec l'enseignement supérieur[22]

4.8 La place de la Communauté.

MandrakeSoft a toujours veillé à renforcer la communauté Open Source, qui favorise par ses contributions le développement continu de l'entreprise et de son offre logicielle, en particulier parce qu'il y a tout à gagner d'une communauté plus soudée et mieux informée.

Née de la communauté, Mandrake met régulièrement en place des actions qui s'inscrivent dans une logique communautaire. Cette logique communautaire, chère à tous les collaborateurs de l'entreprise, vise à :

– *Contribuer à la consolidation de la communauté*[23] .

[19]AXA Placement Innovation, Viventures (Groupe Vivendi), Groupe Iliad, BBS Finance, France Innovation (ABN AMRO), Azeo Ventures (Groupe Lazard)...

[20]distributeurs : ABC Analog, ACI, Art Net, CHS/Métrologie, Editions Profil, ID Pro, Info Networks, Isyotech, Italsel, Kangaroot, Lehmanns, Linux Discount, Macmillan USA, MCD2, Prisma Opentech, Softline ; et offreurs de services : Alcôve, Hewlett Packard, IBM Global Services, IdealX, Savoir Faire Linux, Sun pour la formation.

[21]Alpha Processor Inc., Arkeia, AR Systèmes, ASL, Aurora, Bull, Hewlett Packard, IBM, Matra Datavision (filiale Aérospatiale Matra), MaxSpeed, Quadratec, Solsoft, Sun Microsystems...

[22]Notamment l'ENSTA dans le cadre d'un projet d'enseignement et pour le développement de logiciels Open Source.

[23]MandrakeSoft apporte son soutien à de nombreux organismes et projets de Logiciel Libre : FSF Europe, GNOME Foundation, KDE League, KDE, Plex86, Bastille Linux, Prélude, RpmLint, Urpmi, PHPNuke, Linux Kernel Development.

  - Implication permanente au sein des organisations et mouvements communautaires (FSF, Linux Standard Base, KDE League, GNOME Foundation...)
  - Soutien actif de projets et de contributeurs open-source indépendants (Gnome, KDE, Kernel, KOffice, Plex86).
  - Mise en place d'outils internet au service de la communauté (hébergement de sites, MandrakeExpert, PHP-Nuke).
- *Agrandir la communauté.*
  - en développant des partenariats pour permettre la diffusion massive du Linux
  - en appuyant les actions des LUGs (Groupes d'Utilisateurs Linux) et des organisateurs d'événements autour de la promotion du logiciel libre
  - en favorisant l'accès à la connaissance au plus grand nombre grâce au logiciel libre (projets contre la fracture numérique ou *Digital Divide*)
  –
- *En interne,* des gestes concrets témoignent de l'attachement à ces valeurs
  - choix d'un mode de développement ouvert du produit avec Cooker[24]
  - mise systématique sous licence GPL de nos développements
  - rachat d'une société pour mettre en GPL les licences d'un produit phare (Plex86) au bénéfice de la communauté
- *LUGs* Enfin, MandrakeSoft établit les contacts locaux avec des groupes d'utilisateurs linux sur tous les continents

L'ensemble des sites Web Mandrake (plusieurs URLs signalées plus haut) participent au renforcement du sentiment de communauté, et l'un d'entre eux :

http://www.mandrakeforum.com/

est particulièrement dédié à la circulation des informations, liées à la distribution Mandrake ou plus largement à toute la communauté.

De la même manière,

http://www.mandrakeuser.com/

permet à l'utilisateur Mandrake de ne pas rester isolé, qu'il souhaite ou non être en contact avec les autres (sections de documentation en ligne et section des discussions en ligne).

Concrètement, cette communauté autour de Mandrake est composée à la fois de personnes physiques et groupes d'utilisateurs (le sens premier de la communauté Linux) et d'entreprises partenaires.

4.9 Stratégie.

La stratégie de MandrakeSoft repose sur la promotion de Linux auprès de tous les utilisateurs, du néophyte au plus confirmé, du particulier au professionnel, en élaborant une offre complète et personnalisée de services autour de sa gamme de produits Linux.

Cette stratégie consiste également à défendre des valeurs : la qualité (notamment les utilitaires d'installa-

[24]Reprenant le principe de l'Open Development qui a fait le succès de Linux, MandrakeSoft met à la disposition des programmeurs indépendants et des utilisateurs volontaires une version pilote de ses prochaines distributions, baptisée Cooker.

Plusieurs centaines de développeurs bénévoles ont été séduits par ce projet et par le concept de mise en commun d'idées et de programmes ayant pour objectif de construire la distribution la plus performante. Ce type de projet est une preuve de plus de la volonté et la capacité à rassembler des membres de la communauté Linux.

tion, de configuration, la détection des ajouts de périphériques à chaud, l'interface ergonomique) et l'apport à la communauté. Ces valeurs sont empreintes de légitimité et de sincérité[25] , ce que ne porte pas beaucoup d'autres entreprises opportunistes qui ont voulu «faire dans le Libre» (parasites) et ont rapidement abandonné.

Dans les mois à venir, MandrakeSoft souhaite pouvoir mettre en place un modèle de type SAP.

4.10 Modèle économique.

4.10.1 L'introduction en bourse.

http://www.boursorama.com/graphiques/graphique_histo.phtml?symbole=1r04477-OTC

Ce qui suit est le message (légèrement reformaté) du président Jacques Le Marois envoyé à la communauté juste avant l'introduction en bourse de juillet 2001 pour expliquer pourquoi ce mouvement était nécessaire. Il n'a pas été inutile ; on trouve encore maintenant de nombreux détracteurs qui estiment qu'une telle introduction en bourse a été une trahison vis-à-vis de la communauté. Bizarrement, ces détracteurs sont souvent en France, terre de départ de MandrakeSoft.

> «Née en 1998, MandrakeSoft a souvent été citée dans la presse informatique comme l'un des plus beaux exemples de "success story" Linux. En moins de trois ans en effet, cette société a réussi son premier objectif, ambitieux, de conquête d'une des plus larges bases utilisateurs dans le monde Linux, remportant au passage les récompenses les plus prestigieuses de la profession et la reconnaissance unanime des utilisateurs et des développeurs.
>
> Après cette première étape, un autre projet tout aussi ambitieux a été relevé : transformer un nombre croissant d'utilisateurs en acheteurs de produits Mandrake Linux et de services associés. MandrakeSoft est devenu en effet l'un des leaders du monde Linux, notamment aux États-Unis où, d'après les chiffres de PC Data, elle s'affichait numéro un des ventes de produits Linux au premier trimestre 2001. Un nombre croissant de grandes entreprises font désormais appel à ses services de développement sur mesure.
>
> Le formidable potentiel dont dispose MandrakeSoft commence à peine à être exploité. Plusieurs leviers de développement vont contribuer à accélérer le développement de la société afin d'atteindre le double objectif de croissance des revenus et de retour à la rentabilité :
>
> – l'élargissement de l'offre avec la vente de nouvelles solutions d'infrastructures à destination des entreprises
> – l'accroissement des canaux de distribution indirects à l'international et des ventes directes en ligne
> – le développement de l'offre de services traditionnels - formation, support technique et consulting - auprès des nombreuses entreprises qui utilisent le système d'exploitation Mandrake Linux
> – l'exploitation de nouveaux services en ligne dédiés aux technologies de l'information Open Source (support, place d'échange)
>
> L'introduction en Bourse de MandrakeSoft a pour objectif de financer cet important programme de développement.»

La totalité des 688 480 actions ont été souscrites au prix de 6,2 euros par action, au titre du Placement et de l'Offre à Prix Ferme. Le total des fonds levés ressort à 4,3 millions d'euros. Les actions nouvelles

[25]«Sincérité», c'est pourquoi nous hésitons à rappeler dans cette section Stratégie que le recrutement de fi gures du Libre (voir infra.) à travers un mécénat parfois à 100 % offre à la fois un avantage concurrentiel et l'augmentation de l'image de marque.

représentent 20,28 % du capital après émission.

4.10.2 Mandrake début 2002.

MandrakeSoft clôture son exercice fiscal au 30 septembre 2001. Les résultats consolidés présentés ont été arrêtés par le conseil d'administration réuni en séance le 12 décembre 2001.

au 30/09/01 (EURM)	1999	2000	2001
Chiffre d'affaires	0,6	3,0	3,6
Résultat d'exploitation	0,1	-6,5	-13,4
Résultat net	0,1	-6,8	-13,6

Tableau 4.1 **—** Tableau et commentaires extraits de companynews, à partir d'un communiqué de Mandrake début 2002.

Le chiffre d'affaires consolidé 2000-2001 ressort à 3,6 ME en croissance de 18,9 %.. Cette performance est toutefois légèrement inférieure de 0,7 ME aux prévisions données lors de l'inscription au Marché Libre en raison du ralentissement des ventes "retails"[26] (73 % du CA) en Europe et aux États-Unis (65 % du CA) amorcé en juillet et accentué après les événements de septembre 2001. En revanche, la vente directe en ligne au travers du site MandrakeStore (9 % du CA) et les Services pour les entreprises (16 % du CA) ont atteint leur objectif, validant ainsi la nouvelle stratégie commerciale du groupe. Noter une division par deux des frais généraux et des frais de personnel entre avril et novembre

Après un premier semestre 2000-2001 (septembre-avril) stable, MandrakeSoft a su accélérer sa croissance au second semestre avec une progression du chiffre d'affaires consolidé de 45 % sur cette période par rapport l'exercice précédent et ce, tout en réduisant progressivement ses effectifs.

Les résultats sont, comme prévu, fortement déficitaires mais conformes aux prévisions d'introduction. Le premier semestre a été marqué par le manque de résultats de la stratégie de diversification vers l'*e-learning* menée par le management international recruté fin 2000. Les lourdes dépenses (hommes / R&D / marketing), liées à cette politique, ont mécaniquement pénalisé les résultats du groupe conduisant MandrakeSoft à se recentrer sur ses fondamentaux autour de Linux en mettant en place des mesures drastiques de réduction des coûts et d'amélioration des marges au second semestre :

- Changement du management et arrêt des diversifications *e-learning*,
- Réduction d'1/3 des effectifs et de 57 % de la masse salariale,
- Réduction de la moitié des frais généraux,
- Lancement d'activités plus rémunératrices (e-commerce, services)

Ces mesures portent leur fruits puisque, tout en accélérant sa croissance sur le 2ème semestre, MandrakeSoft a pu diviser par plus de deux ses frais généraux et frais de personnel mensuels entre avril et novembre. Ces derniers ressortent fin 2001 à EUR0,7 M. Déjà perceptible sur l'exercice clôturé au 30 septembre 2001, l'impact sur les résultats du groupe ne sera sans doute visible qu'en 2001-2002.

Dans ces conditions, MandrakeSoft prévoit d'atteindre l'équilibre d'exploitation lors du dernier trimestre de l'exercice 2001-2002.

[26] à travers la grande distribution, la Fnac...

Par ailleurs, MandrakeSoft a mandaté la banque KBC Securities pour une augmentation de capital réservée auprès d'investisseurs financiers ou de partenaires industriels. L'objectif de cette levée de fond est d'accélérer le développement de MandrakeSoft sur le marché des entreprises soit via le recrutement d'équipes dédiées soit par croissance externe.

Le nombre de titres en mars 2002 est de 3 410 000 pour une capitalisation boursière totale de 10,230 millions d'euros.

4.10.3 Création du club des utilisateurs.

Le club des utilisateurs est une nouvelle source de revenu depuis fin 2001.

– *Club des Utilisateurs de Mandrake Linux*, avec plus de 2000 personnes inscrites, et 10000 d'ici moins d'un an. Le premier niveau d'adhésion au club coûte moins de EUR6
– *Mandrake Linux Corporate Club* Pour les entreprises utilisant Mandrake Linux pour ses activités quotidiennes, l'adhésion apporte une reconnaissance particulière et des privilèges exclusifs (adhésion de EUR3000 ou plus).

Depuis quelques semaines Mandrake lance un appel à tous les utilisateurs de ses produits qui n'ont pas encore rejoint le club. En effet, malgré une couverture de presse élogieuse, et un grand engouement pour les produits Mandrake et encore plus pour ses dernières sorties, innovatrices, et des revenus en forte croissance, les coûts de développement et les coûts liés aux services Internet offerts à la communauté Mandrake Linux ne sont pas encore couverts. Il est prévu d'atteindre l'équilibre financier à fin 2002, mais ce sera peut-être trop tard.

MandrakeSoft met la communauté devant ses responsabilités : jusqu'à quel point peut-on profiter toujours gracieusement de produits et de services de qualité ? Et la question doit être résolue rapidement, ce qui paraît d'autant plus facile que les coûts d'adhésion sont faibles, et que des privilèges supplémentaires sont alors offerts. Pourtant, la communauté reste partagée entre ceux qui comprennent et trouvent naturel cet appel, et ceux qui estiment qu'une entreprise du Libre ne devrait pas emprunter de tels chemins jugés mercantiles.

4.11 Une histoire d'hommes et de femmes.

La mésaventure de Mandrake en 2001 qui a conduit à une baisse des effectifs démontre qu'un bon choix managerial est, dans les entreprises du Libre plus qu'ailleurs, peut-être, primordial. Si une éthique n'est pas présente, si la culture du Libre est absente, l'entreprise n'est plus viable[27]. La direction de MandrakeSoft s'attache à présent très fortement à recruter disposant déjà de cette culture, où à même de pouvoir l'acquérir. Au contact les uns des autres, les managers sont «convertis» peu à peu...

Les informations suivantes sur l'équipe managérial sont en grande partie reprises de :

http://www.mandrakesoft.com/company/about/executives

Elles sont à compléter avec les profils «développeurs issus de la communauté» proposés dans la dernière partie de ce document.

Jacques LE MAROIS, Président directeur général, Co-fondateur.

[27]C'est ce qui a pu se passer avec Linuxcare.

Diplômé de l'École Normale Supérieure et du Collège des Ingénieurs, Jacques Le Marois est un pionnier des nouvelles technologies en France. Après l'aventure Internet, il s'intéresse au marché Linux et favorise son essor auprès des entreprises. Le 10 octobre 1998, il organise une Linux Party nationale, une immense démonstration réunissant 35 villes participantes et plusieurs milliers de visiteurs.

Jacques Le Marois a l'habitude de mener des projets d'envergure auprès des sociétés. À 26 ans, il participe au lancement d'une nouvelle entité du Groupe Danone. Puis, en 1997, il réorganise et met en oeuvre un système de reporting sur l'occupation des bureaux virtuels d'Andersen Consulting à Paris et à Amsterdam. Il est l'un des co-fondateurs de MandrakeSoft.

Gaël DUVAL, Co-fondateur, créateur de Linux-Mandrake.

Diplômé d'un DESS «réseaux et applications documentaires» de l'Université de Caen, Gaël Duval est un pionnier de Linux en France : il a assuré la promotion de KDE sur un site spécialisé, créé une liste de diffusion «LIBRE» reconnue par les spécialistes du domaine, participé à des traductions de textes de référence sur Linux, conçu le premier site du monde à proposer l'intégralité d'un disque au format MP3 sur Internet en fin d'année 1997...

Gaël Duval est quelqu'un qui a non seulement des compétences techniques poussées (nécessité pour avoir pu construire la première distribution Mandrake) mais également les qualités propres à tisser des liens dans le monde entier. C'est en s'inspirant de ce double profil que lui vient l'idée, en 1998 (alors qu'il est âgé de 25 ans), de créer Linux-Mandrake. Le concept de base consiste à associer la distribution la plus performante du marché avec le meilleur environnement graphique possible. S'appuyant sur son réseau de contacts internationaux, il assure la promotion et la diffusion des premières versions via Internet.

Frédéric BASTOK, Directeur technique, Co-fondateur.

Frédéric Bastok est un ingénieur des Travaux publics diplômé de l'ESTP. Il a été en charge de la section Linux du magazine PC Expert. Il a dirigé ensuite le département Support de la société OperaSoftware.

Depuis la création de MandrakeSoft, Frédéric Bastok a un rôle prépondérant dans les relations avec la communauté Open Source, ainsi que dans la définition des innovations technologiques et logicielles apportées à la distribution Linux-Mandrake.

Marc PELTIER, Chief Operating Officer.

Diplomé de l'Ecole Supérieure de Commerce de Paris (ESCP), il est titulaire d'un MBA. Tout d'abord, Contrôleur de Gestion chez Philip Morris, il est ensuite devenu Responsable de Projet pour un projet pilote européen. Puis, il a rejoint la société Cimad Conseil (détenue par IBM) comme Directeur de Projet, spécialisé dans la mise en place de projet ERP, et a participé au développement de l'Offre de Services. À la suite de l'intégration de Cimad Conseil au sein d'IBM Global Services, il fut promu Responsable Grands Comptes pour le secteur industrie au sein de l'Offre de Services ERP.

À son entrée dans l'entreprise, en février 2000, et avant d'accéder au poste de Chief Operating Officer, Marc Peltier était Directeur des Services de MandrakeSoft.

Daniel MORALES, Vice-Président, MandrakeSoft Amérique du Nord.

Ingénieur Système et Chimie diplômé de l'Instituto Tecnologico de Monterrey (Mexique), et titulaire d'un MBA de l'Université de Californie (Los Angeles), Daniel Morales a pour mission de diriger les opérations et implémenter de nouvelles initiatives pour MandrakeSoft aux États-Unis, au Canada et en Amérique Latine.

Avant de rejoindre MandrakeSoft, quelques mois seulement après la création de l'entreprise, Daniel Morales était Directeur du Développement Économique et des Programmes d'Assistance aux Entreprises

dans la région de Los Angeles. Il a été responsable de l'administration et de l'obtention de 125 Millions de dollars en prêts et programmes d'assistance financière pour de nombreuses entreprises dans cette même région. Il a aussi fondé, lancé et dirigé de nombreuses startups dans un cadre multinational.

Barry COCHRANE, Directeur international des Ventes et Services, Région Europe.

En tant que Directeur des Ventes et des Services, Barry Cochrane est responsable de la mise en place et la direction des meilleures pratiques et du plan de développement optimal pour MandrakeSoft en Europe. Barry Cochrane a plus de trente ans d'expérience dans le monde de l'informatique et une expérience étendue dans l'établissement de solutions technologiques dans les entreprises et sur les marchés internationaux. Cette expérience inclut le processus entier d'entrée en bourse pour des entreprises telles Sybase et Pure Software.

Avant de rejoindre MandrakeSoft, Barry Cochrane a d'abord été Responsable Européen pour Numéga chez Compuware puis lancé Linuxcare en Europe. Barry Cochrane débuta sa carrière dans la Royal Airforce en qualité d'analyste systèmes puis eut différentes positions successivement chez Cullinuet, Honeywell et Admiral.

4.12 Interactions.

On l'a dit dans la section Communauté, MandrakeSoft soutient de nombreux organismes et projets Logiciel Libre. le recrutement de figures du Libre à travers un mécénat parfois à 100 % offre à la fois un avantage concurrentiel et une augmentation de l'image de marque de l'entreprise. Il faut noter, du reste, qu'à l'heure actuelle tous les développeurs-phares du Libre ont été approchés par des entreprises, et qu'il en reste peu «sur le marché».

4.12.1 Free Software Foundation (FSF) Europe.

Responsable(s) du projet : Richard Stallman (Père fondateur du projet GNU et de la Free Software Foundation). Bernhard Reiter, Peter Gerwinski, Werner Koch et Georg C. F. Greve.

Date de création : Hiver 2000

Description : Les objectifs de la FSF Europe comprennent : la coordination des initiatives de la FSF en Europe, la fourniture d'infrastructures pour les projets liés au logiciel libre (particulièrement le projet GNU), et la mise en place d'un centre de compétence à destination des hommes politiques et de la presse.

L'aide de MandrakeSoft : Le 28 décembre 2000, MandrakeSoft a fait don EUR2500 à la Free Software Foundation Europe, l'organisation soeur de la FSF américaine. Cette aide permettra à la FSF Europe de faire face aux dépenses juridiques contractées en janvier 2001.

Avancement du projet : La FSF Europe vient d'être créée. Son activité prendra un rythme soutenu en Allemagne, France, Suède, et Italie au cours du premier semestre 2001. D'autres pays, comme l'Angleterre, la Belgique, ou les Pays-Bas doivent suivre peu après.

Pour plus d'information : http://www.fsfeurope.org/

4.12.2 GNOME Foundation.

Date de création : 2000

Description : La Fondation GNOME est une association à but non lucratif destinée à faire progresser le projet homonyme. Le projet GNOME a donné naissance à un environnement graphique complètement gratuit et simple d'utilisation, ainsi qu'à une [application framework] sur Linux et d'autres systèmes d'exploitation

basé sur Unix. GNOME fait partie du projet GNU. La Fondation GNOME apportera le soutien financier, juridique et organisationnel nécessaire au projet. Elle aidera aussi à fixer sa philosophie et sa feuille de route.

L'aide de MandrakeSoft : MandrakeSoft a rejoint le Bureau consultatif de la Fondation GNOME le 14 décembre 2000. La distribution Linux-Mandrake inclut l'environnement GNOME depuis ses toute premières versions, et il est récemment devenu l'environnement graphique préféré des utilisateurs Mandrake. MandrakeSoft apportera un sponsor financier aux développeurs GNOME pour leur permettre de se focaliser sur le développement des projets GNOME.

Pour plus d'information : http://www.gnome.org/

4.12.3 KDE League.

Date de création : 15 novembre 2000

Description : La KDE League a été créée grâce à une alliance entre des groupes industriels très importants et les développeurs KDE, destinée à favoriser la promotion, la distribution et le développement de KDE. La League se focalisera sur la promotion de l'utilisation du bureau Open Source alternatif très avancé qu'est KDE sur les PCs, stations de travail, et objets mobiles, et ce auprès des entreprises et des particuliers. La League portera aussi ses efforts sur l'incitation des développeurs tiers à développer pour KDE.

L'aide de MandrakeSoft : En tant que membre fondateur de la KDE League, MandrakeSoft s'est associé avec d'autres acteurs majeurs du secteur informatique pour fournir à KDE le soutien financier, moral et publicitaire nécessaire.

Pour plus d'information : http://www.kdeleague.org/

4.12.4 KDE.

Responsable(s) du projet : David Faure, Mosfet (Daniel M. Duley)

Date de création : Octobre 1996

Description : KDE (K Desktop Environment) est un puissant environnement graphique Open Source pour stations de travail Unix. Il allie la facilité d'utilisation, des fonctionnalités très modernes et un design époustouflant à la supériorité technologique d'Unix. KDE est un projet Internet réellement Open Source dans tous les sens. KDE a développé une cadre de développement [development framework] Unix de très haute qualité, qui permet la création rapide et efficace d'applications. KDE est le fruit du travail de plusieurs centaines de développeurs venant de plus de trente pays.

L'aide de MandrakeSoft : MandrakeSoft a recruté les développeurs David Faure, Laurent Montel, et Daniel M. Duley (Mosfet) afin qu'ils se consacrent au développement de KDE-2. MandrakeSoft héberge aussi le site Webcvs dédié à KDE.

Avancement du projet : Le 31 janvier 2001, l'équipe KDE a annoncé la disponibilité de KDE 2.1-beta2, un puissant environnement graphique, modulaire et optimisé pour l'Internet. KDE 2.1 est la deuxième version majeure de la série 2.

Pour plus d'information : http://www.kde.org/

4.12.5 Plex86.

Responsable(s) du projet : Kevin Lawton

Date de création : Août 1999

Description : Le but du projet Plex86 est de créer une application Open Source évolutive de virtualisation, qui permettra aux utilisateurs de PC et de stations de travail de faire fonctionner en même temps plusieurs systèmes d'exploitation. Plex86 fera fonctionner de manière aussi native que possible une grande partie du système d'exploitation et des applications, le reste étant émulé par le système de de virtualisation du PC.

L'aide de MandrakeSoft : MandrakeSoft a recruté Kevin Lawton, le responsable du projet Plex86, en été 2000.

Avancement du projet : La première application utilisateur est attendue pour avant l'été 2001

Pour plus d'information : http://www.plex86.org/

4.12.6 Bastille Linux.

Responsable(s) du projet : Jay Beale, Yoann Vandoorselaere

Date de création : Décembre 1999

Description : Le projet Bastille Hardening System vise à étendre le degré de sécurisation du système d'exploitation Linux. Il fonctionne pour le moment avec les distributions Red Hat et Mandrake. L'équipe de développement souhaite fournir le système le plus sécurisé qui soit, tout en restant peu contraignant pour l'utilisateur. Le projet est conduit par Jon Lasser (principal coordinateur) et Jay Beale (développeur en chef). Un grand nombre de développeurs, de beta-testeurs et de concepteurs sont impliqués. Bastille Linux a été conçu en visant plusieurs objectifs : exhaustivité, éducatif, communautaire.

L'aide de MandrakeSoft : MandrakeSoft a recruté Jay Beale (développeur en chef) le 14 novembre 2000, pour diriger sa grandissante équipe dédiée à la sécurité.

Avancement du projet : Bastille Linux continue son développement avec une nouvelle interface utilisateur et des mises à jour pour des fonctionnalités pare-feu, XINETD, et DNS optimum.

Pour plus d'information : http://www.bastille-linux.org/

4.12.7 Prelude.

Responsable(s) du projet : Yoann Vandoorselaere

Date de création : 1998

Description : Prelude est un Système de Détection d'intrusion réseau. Il est composé des programmes Prelude et Prelude Report. Le premier sert à la capture des paquets et leur analyse via plugins, le second, à rapporter les attaques d'une façon lisible par l'utilisateur. Ceci permet d'avoir les rapports sur une autre machine. Les autres caractéristiques importantes de Prelude sont une pile de défragmentation IP et des facilités fournies aux plugins pour avoir un état persistant.

L'aide de MandrakeSoft : Yoann Vandoorselaere, responsable du projet, a été recruté par MandrakeSoft en Février 1999, dans le but d'apporter à Prelude toute l'aide nécessaire.

Avancement du projet : La version 0.4.1 de Prelude vient de voir le jour.

Pour plus d'information : http://www.linux-mandrake.com/prelude/

4.12.8 RpmLint.

Responsable(s) du projet : Frédéric Lepied

Date de création : October 2000

Description : RpmLint sert à vérifier les erreurs communes lors de création de paquetages RPM. Les paquetages sources et binaires peuvent être vérifiés.

L'aide de Mandrakesoft : MandrakeSoft a recruté en décembre 1999 Frédéric Lepied, le responsable du projet, et lui apporte tout le support nécessaire au développement du projet.

Avancement du projet : RpmLint en est à sa version 0.29.

4.12.9 Urpmi.

Responsable(s) du projet : François Pons, Pascal Rigaux (Pixel)

Date de création : Mai 1999

Description : Urpmi permet à un utilisateur autorisé d'effectuer des paquetages depuis plusieurs média. Ses fonctionnalités clés sont la gestion automatique des dépendances et la recherche d'expression dans la liste des fichiers de tous les paquetages gérés.

L'aide de MandrakeSoft : Pascal Rigaux, le créateur du projet, a été recruté en juillet 1999 par MandrakeSoft, qui lui apporte depuis cette date tout le soutien nécessaire pour développer le projet.

Avancement du projet : Urpmi en est à sa version 1.5.

4.12.10 PHP-Nuke.

Responsable(s) du projet : Francisco Burzi

Date de création : été 2000

Description : PHP-Nuke est une application-clé à la manière de Slashdot, qui permet à tout un chacun de créer facilement pour le Web des fonctionalités de news sur un mode communautaire. Le but de PHP-Nuke est de fournir un site automatisé permettant la diffusion d'actualités et d'articles, enrichi d'un système de compte utilisateur. Chaque utilisateur peut publier des commentaires sur les articles. Disponible en 23 langues, PHP-Nuke a déjà été téléchargé par plus de 140 000 utilisateurs en six mois, et est utilisé par des centaines de site à travers le monde. PHP-Nuke est entièrement écrit en PHP, il nécessite un serveur Web Apache, le langage PHP et une base de données MySQL.

L'aide de MandrakeSoft : En rejoignant l'équipe MandrakeSoft, Francisco Burzi, le responsable du projet PHP-Nuke ne bénéficie pas seulement d'une aide financière, mais aussi d'un support technique total incluant hébergement Web, sites miroirs pour le téléchargement ; il pourra aussi tirer partie du réseau de communication MandrakeSoft pour étendre la notoriété de PHP-Nuke auprès des utilisateurs du monde entier.

Avancement du projet : La dernière version de PHP-Nuke est la 4.4, disponible depuis février 2001.

Pour plus d'information : http://www.phpnuke.org/

4.12.11 Développement du Kernel Linux.

Responsable(s) du projet : Linus Torvalds

Date de création : Septembre 1991

Description : Le kernel est la composante fondamentale du système d'exploitation Linux, programmée à partir de rien par Linus Torvalds, avec l'aide d'une équipe peu structurée reliée par l'Internet. Il a toutes les caractéristiques d'un vrai système d'exploitation moderne. Le kernel est distribué sous la General public Licence.

L'aide de MandrakeSoft : Un des développeurs clés du Kernel, Jeff Garzik, a été recruté par MandrakeSoft

en octobre 1999. MandrakeSoft fournit à Jeff Garzik toute l'aide financière et technique nécessaires pour faciliter sa contribution continue au kernel pour le plus grand bien de la communauté Linux. Philip Rumpf et Juan Quintela ont été aussi recrutés pour contribuer au développement du kernel.

Avancement du projet : La toute dernière version du kernel est la 2.4.

Pour plus d'information : http://www.kernel.org/

Les sociétés de services en logiciels libres : l’exemple de Makina Corpus.

Aymeric Poulain-Maubant,
Jipo Interactive, Aymeric.PoulainMaubant@jipo.com.

5.1 Makina Corpus

– Personnes interviewées : Jean-Pierre Oliva (directeur général), Philippe Julien, Mose
– Site de référence : http://www.makina-corpus.com/

Makina Corpus est une SS2L (Sociétés de Services en Logiciels Libres) qui fournit des services et des applications Internet et télécom. Elle est implantée en France, au Royaume-Uni et en Espagne.

Elle a été créée par un ingénieur télécom ayant une longue expérience du conseil. Il a des affinités pour le libre mais plus en tant qu'utilisateur que développeur. Il est toujours présent comme directeur général et pilote les grands orientations stratégiques.

5.1.1 Activité

Makina Corpus s'articule autour de 4 pôles :

– *Développement* de solutions web fiables et performantes, et de développement d'applications dans des domaines allant des systèmes de télécommunication aux applications de téléphonie mobile, des intranets aux sites e-commerce, de l'internet mobile aux sites m-commerce,
– *Conseil* stratégique, techonologique, assistance à maîtrise d'ouvrage et gestion de projets, études des besoins, audits et spécification de cahiers des charges...
– *Formation* sur les logiciels libres, comme par exemple celle destinée aux centres d'appels des fournisseurs d'accès Internet, confrontés de plus en plus à des utilisateurs linux. Ces formations reposent sur une veille continue (elle-même prestation possible),
– *Services*, installation et gestion de parcs, migrations, administration à distance, infogérance, assistance aux utilisateurs...

Outre les cibles affichées que sont les prestataires Internet (fournisseurs d'accès, d'hébergement...) et les prestataires télécoms (et en premier lieu les opérateurs), Makina vise également les agences de communication et *web agencies* susceptibles d'utiliser les logiciels de gestion de contenu de sites web développée par icelle. Par ailleurs, l'élargissement des types de clientspeut s'effectuer via des consultants Makina ayant été placés chez le client et détectant (avec l'accord du client) les besoins des clients de ce client.

Le logiciel libre est avant tout un choix d'outils et de méthodes. Makina Corpus a fait ce choix car il permet une ouverture permanente sur les nouvelles technologies, garantit l'indépendance dans les choix techniques et permet de se concentrer sur ce qui est le plus important : apporter la réponse la plus adaptée aux besoins des clients. C'est pourquoi même si Makina effectue des prestations —on dira classiques— d'installation sur site, de maintenance d'intégration hétérogène, d'adaptation de logiciels libres pré-existants et de développement à façon, tâches que l'on imagine dévolues à des informaticiens, sa valeur réside aussi dans les prestations de conseil et d'assistance (comme par exemple l'intermédiation entre entreprises et développeurs indépendants), une démarche qui va faire le pont entre le monde parfois chaotique du logiciel libre et celui de l'entreprise.

Deux types de logiciels font partie du «portefeuille» de Makina Corpus : ceux qui viennent de l'extérieur comme Zope (et d'autres qui sont continuellement testés puis ajoutés aux compétences de Makina), et ceux qui sont produits en interne, et pour lesquels on voit combien ils relèvent des principes de Raymond [1998a] : «un développeur travaillement d'autant plus facilement sur un outil dont il a personnellement besoin». Noter que ces logiciels s'enrichissent au fur et à mesure des implémentations chez les clients de Makina. Nous en détaillons quelques-uns à présent.

Zope

Zope est un serveur d'application et de contenus, soutenu par une importante communauté. Il permet notamment de donner la possibilité à des personnes non-techniques de gérer des sites complexes.

Zope n'est pas développé par Makina Corpus, et aucun Makiniste ne participe pour l'instant de manière soutenue à son développement.

CCMS

Le Collaborative Content Management System est un outil de gestion de site web par interface web, écrit en php, fondé sur un stockage de fichiers en xml, avec une authentification modulaire ldap, mysql ou fichier texte. Ses caractéristiques sont de donner accès via une interface utilisateur ergonomique à tous les éléments constituant un site web évolué, multi-domaines et multilingue : système de gabarits personnalisables, styles de texte enrichis, gestion de variables en fonction du contexte. Cet outil est destiné à faire collaborer les différents intervenants d'un site : le design, le contenu, le référencement peuvent être gérés de front.

Son auteur principal est Mose.

CCMS ne s'inspire pas directement de logiciels propriétaires, bien qu'il en existe néanmoins ayant des objectifs semblables. Il s'inspirerait plutôt d'autres outils libres, comme *cWriter* (voir infra.), *spip* ou *dacode*[1], outils ayant émergé à la même époque que les premiers brouillons de Mose.

Comme cWriter (on sent la patte de l'auteur), l'originalité de ce logiciel vient de la combinaison d'éléments classiques par ailleurs, l'agencement de l'interface créant un contexte spécifique à cette application. Les points forts du CCMS sont l'historisation infinie, l'usage d'un *rich text*[2] à l'instar de *wikiwiki*[3] ou de spip, mais paramétrable par l'utilisateur, un très profond niveau de paramètres pour influer sur les mécaniques de visualisation du site...

Le principe du CCMS est issu d'un pratique quotidienne de la fabrication de sites depuis les débuts du web. Les solutions mises en place correspondent à des besoins de production immédiats. L'ancêtre du CCMS a été développé quelques mois avant l'existence de Makina Corpus, en février 2001. Il était alors destiné à deux usages distincts, la gestion du site claranet[4] d'une part, et celle du site de Cardionews [Mose] (association de cardiologues) d'autre part. Celui pour claranet n'a jamais abouti, mais celui de cardionews a été finalisé, et son exploitation est gérée à présent par keo.net[5]. Les premiers sites commandés a Makina Corpus ont bénéficié du gabarit de gestion, qui s'est affiné au fil des sites, en se confrontant à différentes contraintes (*frames* ou non, besoin de formulaires personnalisés, maquettes atypiques, multitude de sites avec une même maquette, etc.).

Le *packaging* en vue de sa diffusion en libre a commencé en septembre, pour s'achever en décembre 2001. La mise en ligne, début janvier 2002 a été l'occasion de prises de contacts positives, et Makina Corpus a bon espoir pour la phase d'appropriation qui est en cours. (Voir également section Communauté).

[1] Respectivement http://savannah.gnu.org/projects/cwriter, http://www.spip.org/ et http://www.dacode.org/

[2] L'utilisateur n'est pas obligé de connaître les codes HTML pour le gras, par exemple, il lui suffi t d'entourer le mot qu'il souhaite en ***gras*** par des étoiles, ce qui ne romp pas le fi l de saisie du texte et traduit bien visuellement l'idée de graissage.

[3] http ://c2.com/cgi/wiki ?WelcomeVisitors

[4] http://www.claranet.fr/ Prestataire Internet où se trouvait Mose avant de rejoindre Makina Corpus. Voir section RH.

[5] Autrement dit, le service associé au logiciel est effectué par un tiers.

cWriter

Le Rédacteur Collaboratif (cWriter est le nom anglais) est un outil en php et mysql qui propose un environnement en ligne de collaboration rédactionnelle pour un groupe de 10 a 100 personnes[6]. Le contexte de rédaction a été agrémenté de fonctionnalités destinées a favoriser la collaboration autour de la rédaction : système rédacteur / collaborateur, planning partagé, annuaire, formulaires de contact, téléchargement de fichiers, banque de liens, accès aux statistiques de consultation, version imprimable, version intégrale de chaque document, système de verrouillage en écriture, inclusion de commentaires dans le corps d'un document, etc.

Son auteur principal est Mose.

Il s'inspire peu ou prou de logiciels propriétaires (Lotus) ou semi-propriétaires (BSCW), plus par les objectifs et usages souhaités que par l'interface et les fonctionnalités particulières et la démarche. Il a été développé suite à des tentatives de travail collaboratif sur ces plate-formes propriétaires qui se sont révélé impliquer de lourdes charges de gestion et une expertise spécifique non capitalisable. À l'époque (milieu des années 90), seul BSCW permettait d'envisager un contexte de travail collaboratif, mais la complexité de son interface a brisé l'appropriation dès les premiers essais. Le temps de développement a été étalé dans le temps, sur des années, au fil des expériences dans des communautés[7] diverses, sans aucun plan stratégique préalable si ce n'est la volonté de construire touche après touche une meilleure fiabilité/fluidité de collaboration.

Son originalité provient de la combinaison de certaines de ses fonctionnalités, classiques par ailleurs, qui en font un produit qui ne ressemble qu'à lui-même. La plupart de ses fonctions n'ont rien d'original, mais leur agencement est le résultat de longues observations des usages. Quelques détails cependant sont moins courants, comme par exemple la conservation de chacune des étapes de saisie pour un retour historique infini. L'intérêt du cWriter réside dans la concentration de fonctionnalités focalisées autour de la formalisation d'un document écrit, mais cet usage peut être detourné, et certains documents sous cWriter pourraient être décrits par ceux qui les fréquentent comme des places de débats, ou des systèmes de sondages.

Le premier jet du RC s'appelait 'el communiante', un outil rapide de rédaction collaborative produit dès la sortie de php3, pendant l'été 98. Le but était de permettre la rédaction du dossier de suivi technique d'un événementiel qu'organisait Wanadoo, et qui avait pour l'occasion embauché les Internautes Associés dont Mose était un sattelite. Un contact à l'époque, Dominique Gros, consultant en Formation à Distance, a beaucoup travaillé à l'aspect fonctionnel et à la mise en place de postes de test. Mais c'était encore trop tôt, et la disponibilité en pointillé de Mose ne lui permettait pas de suivre les délais. Il a été réécrit plusieurs fois, portant le nom d'orc, arc, puis enfin rc.

Il a été deployé pour l'université de Clermont-Ferrand Amberlab [Mose], dans des cercles communautaires de façon expérimentale (Cap-Corse puis Radio-Phare, mais aussi Acuarel[8]), et en interne chez Claranet pour la gestion de la base de connaissances.

En novembre 2000 le code source a été proposé à la communauté et diffusé sur Sourceforge, sous licence GPL. Il y a eu environ 300 chargements depuis, ce qui reste infime, et pourtant considérable (du point de vue de l'auteur qui reçoit autant de commentaires qui peuvent servir à améliorer le logiciel). Il faut noter

[6]Noter que 80 % des matériaux de cette monographie ont été construits à partir d'un RC spécialement dédié. L'auteur y avait posé ses questions, et divers contributeurs ont répondu et même échangé entre eux sur un document en construction.

[7]Prendre ce terme dans son sens premier de groupes de personnes ayant des affi nités.

[8]Voir http://www.radiophare.net/. L'auteur de cette monographie a fait partie de cette petite communauté Acuarel composée de sociologues, économistes, géographes, ingénieurs informaticiens, étudiants et chercheurs, internautes de la première heure et humeurs de temps, s'étant rencontrés sur Internet ou dans des vies antérieures, et qui ont souhaité discuter ensemble en mai 98 du premier tome de Manuel Castells sur la Société en Réseau (Castells [1998]). Acuarel était un acronyme sur «Architecte, Concierge, Auteur, Rédacteur, Éditeur, Lecteur», rôles endossés par les Utilisateurs du RC.

cependant que Mose estime avoir été plus soutenu grâce aux sollicitations de ses proches, qui avaient passé le seuil d'appropriation, que par la ferveur de la communauté libre, ce qu'il explique, avec le recul, dû aux choix d'organisation du développement.

En effet, pour être libre (libéré de l'emprise de son auteur[9]), un logiciel doit avoir en premier lieu un *script* d'installation efficace, qui permettra à d'autres développeurs d'avoir une chance de tester, même si rien ne marche. Dans le cas de cWriter le *script* d'installation a été produit tardivement, et l'appropriation de cet outil par la communauté libre sera tardive. À vrai dire elle commence a peine.

5.1.2 Concurrents

Zope supporte la comparaison avec les logiciels propriétaires et libres (Cold Fusion, MidGard, Enhydra, Interwoven, voire Broadvision et Vignette). Il présente des points forts et des points faibles, notamment ceux liés à Python, mais l'éco-système logiciel dans ce type d'applicatifs reste encore jeune et laisse suffisament de place aux uns et aux autres. Il souffre cepedant d'un déficit d'image (de visibilité) et d'hébergeurs.

CCMS a peu de concurrents dans sa catégorie. Compte-tenu du contexte (une application pour faire des sites web à partir du web), les concurrents (des agences web, par exemple) font peu de publicité car il est souvent facile de s'inspirer du code des autres (quand les fonctionnalités sont en javascript, notamment). Ces logiciels sont donc protégés dans des extranets ou ont leurs composantes principales côté serveur (ce qui est une sorte de propriétarisation).

CCMS manque peut-être de stabilité (de nouvelles fonctionnalités sont ajoutées quotidiennement) et d'un grand nombre d'installations.

Un autre logiciel libre, eevote/Glasnost[10], proposé par Easter eggs, à base de Spip et de python, semble poursuivre les mêmes buts que le RC ou le CCMS (eux-même par certains aspects parfois similaires), même s'il se concentre avant tout sur l'outil de vote. Mose estime que dans ce cas il n'y a pas de rapprochement ou de fusion envisagés. Il n'y a pas non plus de conflit, mais plutôt un réflexe naturel de négation des similitudes pour éviter ce conflit potentiel : it «une sorte de dérivation dans l'axe de développement peut même s'en ressentir, d'un côté comme de l'autre. Mais le Libre tolère (et favorise) la multitude, bien mieux que la voie commerciale pour qui l'effet levier vaut tout (et fait d'autant mieux levier qu'il y a peu de points d'appui).»

5.2 La place de la Communauté

La Communauté, au sens premier du terme, n'est pas encore très présente autour de Makina Corpus. Mais les développeurs de Makina sont issu de la Communauté, ou d'un sentiment de communauté, ou de communautés restreintes ; on en traitera dans la section Ressources Humaines. Et ils activent leurs réseaux communautaire, fréquentent des communautés (Atica...), y cherchent synergies et clients.

Parce que Makina Corpus développe des logiciels libres, la Communauté est une donnée importante : cette dernière doit pouvoir tester, commenter, critiquer, s'approprier, détourner ces logiciels pour que ces derniers se déploient[11]. De plus, les prescripteurs chez les clients de Makina Corpus sont de plus en plus des personnes

[9]Ceci est certainement vrai dans le cas d'un très petit noyau d'auteurs, voire un seul auteur, qui se contentent au début d'un environnement de développement succinct, et nettement moins vrai quand les auteurs sont très nombreux (plusieurs dizaines ou centaines). Le cas typique est celui de la distribution Debian qui absorbe ses auteurs au lieu de se répandre.

[10]http://www.entrouvert.org/rubrics/2

[11]Dans les deux sens : grandissent / mûrissent et se répandent.

ayant vécu les premières heures d'Internet pour le grand public et de Linux (première moitié des années 90), c'est-à-dire issus ou proches de la Communauté.

Makina Corpus commence à proposer un site web spécialement dédié aux développeurs et utilisateurs curieux Mose [2002], sur lequel sont placés les logiciels développés, mais également des exemples d'utilisation de logiciels libres dans les contextes télécoms, prestataires de services internet ou mobilité qui sont des axes majeurs pour Makina.

Cela dit, il n'y a pas là ambition de déployer des services à ls sourceforge, faute de moyens et de ressources, et surtout parce que ces services existent déjà. Et plus que «exister déjà», ces services comme Savannah ou Tuxfamily[12] hébergent déjà des communautés au contact desquelles les échanges favorisés peuvent être fructueux. «[Ce sont des lieux qui permettent] de confronter le produit au terrain, voire de le modifier pour qu'il s'adapte à ce qui se fait, dans un but eugeniste de normalisation naturelle.» Par exemple, pour être accepté sur savannah il y a des conditions à remplir, qui n'avaient pas été remplies la première fois que RC y a été proposée. Mose a du revenir sur l'ouvrage et le mettre aux normes (de savannah) : détails sur la présentation de la licence, absence d'images au format gif. RC a alors été accepté. Ces règles sont un gage de qualité et l'assurance de ne pas être à chaque fois en terrain inconnu pour la communauté. De même, le CCMS a été refoulé alors que RC était accepté, pour des problèmes d'en-têtes de fichiers alors que Mose avait fait attention (suite à la première tentative pour RC). Rien n'est donc jamais acquis pour un développeur, et encore ne s'agit-il pas là des règles pour inclure un produit dans le projet GNU[13]. Il est intéressant du reste de noter que ces règles qui structurent la forme de logiciels structurent également la communauté qui les adopte.

Mais peut-être le site proposé par Makina rencontrera-t-il aussi sa communauté d'utilisateurs...... Rendez-vous l'année prochaine.

5.3 Promouvoir le Libre

5.3.1 La nécessaire promotion

Chez Makina Corpus, l'auteur lui-même (et les développeurs) d'un logiciel libre en assure sa promotion. Il connaît les niches —de la communauté— où son œuvre sera solidifiée par le *'peer review'*, phase que l'on préfère préalable à la diffusion en masse. Mais il est rare qu'un développeur sache parler de ce qui lui est trop proche, et quand on en arrive à la promotion de l'image de la société, et de la valeur de ses éléments, il vaut mieux faire appel à quelqu'un ayant plus le profil marketing. Makina Corpus a ainsi fait appel à des expert de chez Jipo pour faire une reconnaissance actuelle du contexte de communication autour du libre, et a dégagé un plan de bataille avec de nombreux axes tactiques. En revanche, il n'existe pas[14] encore de personne dont le seul rôle soit d'appliquer les recommandations de Jipo.

Plus largement toutes les personnes de Makina Corpus se trouvant dans un contexte approprié (client, conférence, groupe de discussion...) propos doivent être capable d'assurer la promotion de la structure Makina (logiciels, services et savoir-faire). Le client lui-même doit devenir un ambassadeur de l'outil développé pour lui, ce qui se passe naturellement lorsque le développement est bon. Enfin, le fait de participer à des atelier de travail avec des chercheurs de solutions doit être un des credo de la promotion (Atica, séminaire de décideurs...). Ce travail peut être mené de front par deux personnes (un développeur et un non-développeur, commercial et/ou marketing) pour répondre aux différents types d'interlocuteurs que l'on rencontre dans ce genre de manifestations.

[12] Resp. http://savannah.org/ et http://www.tuxfamily.org/.

[13] Dans ce cas la structure de la documentation, la manière de gérer l'installation, l'écriture du code sont régies fortement.

[14] en février 2002. Les choses évoluent rapidement en fonction des profi ls recrutés.

La promotion passe aussi par l'existence et la qualité du support, ce qui est d'autant plus vrai dans le cas de Makina que le support est un de ses axes commerciaux. Un manuel d'utilisation est toujours fourni, mais déployer un des logiciels de Makina dans un contexte particulier nécessite une période de conseil assurée le plus souvent par le développeur lui-même, qui va donc à la rencontre (et à l'écoute) du client et ne reste pas en chambre. Il nourrit en retour les nouvelles versions de son logiciel avec les remarques et les besoins spécifiques de son client.

L'esprit du Libre n'est pas pour autant dévoyé. Les logiciels sont fournis avec ce qu'il faut pour qu'un utilisateur averti et compétent puisse assurer son propre support. Et dans ces cas il n'y a bien sûr pas de retour immédiat en terme de chiffre d'affaire (le logiciel est gratuit et aucun support n'a été vendu). En revanche, un utilisateur averti et compétent aura tendance à s'approprier le logiciel, à l'apprécier et à le recommander, notamment à des utilisateurs moins avertis. Et pour ceux-là, il recommandera le support fourni par Makina. Le retour s'effectue donc à moyen terme (le temps du cycle d'appropriation par le client ambassadeur).

Il faut noter que la promotion du Libre est de moins en moins nécessaire. Depuis quelques années en France de nombreuses associations (pour ne prendre que cet exemple) ont informé les entreprises, les collectivités, l'administration, les medias et le grand public des enjeux du Libre. Il n'est pas rare que les clients demandent spontanément de produire des logiciels libres (ou dans l'esprit du Libre), qu'ils recherchent les coûts modérés (disons : justes) ou la fiabilité et la sécurité. Une des cibles de Makina Corpus, l'administration, suit ainsi des directives gouvernementales de moins en moins officieuses, et les effets bénéfiques (contrats importants de migration du monde propriétaire vers le monde du logiciel libre) se font déjà sentir.

Les clients demandent certes du logiciel libre, mais la licence en tant que tel n'est pas un élément fondamental pour le décideur. Il est vrai que la terminologie «licence» recouvre une réalité économique toute différente dans les deux mondes propriétaires et libres, et dans le cas des grands comptes, le prix de la licence, même élevé, reste marginal par rapport à une intervention-conseil d'un spécialiste. Le grand compte est plus attiré par la notion de «support par une large communauté», qui lui assure la pérénité de ses logiciels (et de ses données). Une petite structure (collectivité, PME) reste sensible aux coûts modérés et à ces licences généreuses.

5.3.2 Licences

Le choix de la GPL de la FSF a été motivé par plusieurs raisons. D'une part la garantie de non-mercantabilité est un argument de vente, pour les clients qui payent les services de développement (cas des demandes d'intégration ou de développement à façon). Comme cette licence garantit que les retouches ultérieures du produit resteront libres, le danger de la captivité par l'éditeur est ecartée.

Ce choix a également été motivé parce qu'elle représente dans la communauté du Libre : un gage d'intégrité, qui peut être considéré comme un facteur favorisant l'appropriation.

5.4 Stratégie

Makina Corpus n'a pas opté pour une stratégie Libre 100 %. Il n'y aucune raison de refuser a priori des clients qui proviennent du monde propriétaire ou souhaitent disposer de produits propriétaires (ce qui est quasi-obligatoire dans le monde des télécommunications). Certaines des prestations de Makina comme la (télé-)maintenance de systèmes informatiques se déroulent dans des environnements fortement hétérogènes.

Les compétences (originales, pourrions-nous dire) sont en revanche clairement articulées autour du Libre.

En tant que producteurs de logiciels, le choix de Makina est résolument le Libre[15]. Il s'agit là de mettre l'accent sur le service, attirer et conserver un client par la qualité du service plutôt que par un verrouillage avec des licences et des mises à jour incessantes... Les développeurs Makina bénéficient également de retours de contributeurs plus nombreux pour améliorer leurs produits, ce qui a accéléré le *time to market* des solutions Makina[16].

En tant qu'utilisateurs de logiciels libres existants, Makina recherche les atouts propres du Libre (qualité des développements disponibles, plus grande stabilité, ouvert aux critiques et donc aux améliorations, bénéficier des évolutions apportées par la communauté), souhaite ne pas être asujetti aux diktats de fournisseurs de logiciels, et avoir entière latitude pour adapter des logiciels existants (ce qui est nécessaire pour les clients télécom au sens large, très gourmands en applications nouvelles et rapidement développées).

Makina incite ses employés à contribuer à des projets libres extérieurs à la société, pour toutes les bonnes raisons qu'on peut imaginer : c'est l'usage, c'est un moyen de suivre les évolutions de ces logiciels, cela permet de proposer et faire valider de nouvelles idées, cela permet de faire connaître Makina par des développeurs, par des clients, c'est une garantie de compétence pour les clients, voire parfois une demande expresse des clients.

Dans la phase actuelle de la société, l'idée est de se créer un portefeuille de logiciels sur lesquels Makina peut afficher ses compétences (conseil et formation, déploiement, développements particuliers et intégration...). Les consultants, sans cesse à l'écoute des clients, repèrent des logiciels qui sont directement utiles, ou peuvent le devenir après adaptation, et les testent. C'est bien sûr toujours cette approche efficace qui sera préférée à celle de développer *from scratch*.

Cette démarche implique de lancer ses antennes dans toutes les directions (à la fois pour trouver, et pour chercher des testeurs critiques), et ce, pour tous les consultants. Dès lors, des économies sur la communication sont effectuées car le tissu relationnel de chacun s'étend très rapidement. Il s'agit d'un cercle virtueux car, plus ce tissu est dense, plus l'appropriation par des tiers d'une solution logicielle développée par Makina peut s'effectuer rapidement, et plus de nouveaux prospects, moins avertis techniquement (ou disposant de moins de temps) et donc friands de prestations de conseil, d'assistance et de support, sont invités par les tiers-premiers-testeurs.

En la matière, s'exposer aux critiques, dès lors que la qualité (et la réactivité) reste au rendez-vous, est bénéfique.

Enfin, dernier avantage souligné de cette stratégie vers le Libre : participer a un projet libre, c'est aussi parfois profiter de la proximité avec des experts pour se former sur un domaine particulier.

5.5 Modèle économique

Makina Corpus se positionne comme société de conseil et SSII et pas comme éditeur de logiciel. Le modèle économique est donc tout simplement celui d'une société de services, et on ne peut pas dire que le Libre apporte ici une dimension nouvelle, surtout quand l'on sait que Makina travaille aussi sur des environnements propriétaires. Tous les types de clients sont visés mais, pour des raisons historiques, le secteur des télécoms, des prestataires internet, et celui des collectivités tient une place très importante aujourd'hui.

[15] Il n'est absolument pas envisagé de développer des logiciels propriétaires alors que dans le même temps Makina vante les mérites du Libre.

[16] Makina étant fortement utilisatrice de logiciels libres pour son propre compte, il y a également une volonté de contribuer pour la collectivité.

Les revenus proviennent du conseil et des prestations de services (analyse des besoins, personnalisation et développement à façon, maintenance, formation...). La mise à disposition des compétences[17] des consultants, en régie notamment, est également possible.

Il est certains que les revenus provenant du conseil mettront plus de temps à se mettre en place que ceux, découlant de besoins immédiats, liés au développement et aux prestations de services.

Makina Corpus n'est pas dédié à un seul logiciel, même s'il y a un parti pris arbitraire en faveur de Zope dans certains choix. Makina peut être considerée comme généraliste, plutôt spécialisée dans les applications web, et notament le travail collaboratif, le développement sous Zope, les systèmes de gestion de contenu web, la mobilité (GPRS, PDA, *wireless LAN*...) et les systèmes d'informations géographiques, et dédiée à des secteurs comme les télécoms, l'industrie automobile ou les collectivités publiques.

Makina vendant des services, la mise en place de logiciels est intégrée à ces services et n'est pas traitée par des tiers. Elle traite directement avec ses clients, sans intermédiaires ni revendeurs.

Le principal investissement a été le renforcement des compétences des employés sur leurs outils de base et la création d'un cadre de développement solide. Encore jeune, la société a surtout investi (son temps) dans son positionnement et sa visibilité, qui commencent à porter leurs fruits.

Au bout d'un an d'activité, Makina Corpus est à l'équilibre. Les perspectives d'évolutions sont excellentes ; alors que la société était en phase de préparation, elle a déjà conquis et satisfait de nombreux clients prestigieux.

La principale difficulté rencontrée concerne les réticences vis-à-vis du logiciel libre dans certains milieux.

Noter que Makina Corpus ne répond pas aux appels d'offres n'émanant pas d'une personne identifiée et connue.

5.6 Une histoire d'hommes et de femmes

Une personne coordonne les fonctions commerciales et la relation avec la clientèle. Il a une petite expérience du libre mais vient plutôt du monde propriétaire. Une autre coordonne l'équipe de développement ; c'est un activiste du libre. Mais le management opérationnel est plus subtil que cela, avec une hiérarchie à gérométrie variable comme on va le voir.

Sous le houlette du directeur général, trois grandes fonctions sont présentes :

1. *Opérationnel* : équipe de développement, missions et projets (cf. infra directeurs de mission)
2. *Support* : marketing, commercial (cf. infra) et bureautique (cette dernière pouvant également êtrevue dans l'opérationnel, en tant que prestation)
3. *Gestion* : administratif et financier et gestion du personnel

[17] Pointues, ce qui doit se vérifi er à l'embauche, et s'entretenir en continu : curiosité du consultant, participation à des projets Libres externes, implications dans des projets-clients innovants.

5.6.1 Organisation

Makina Corpus est organisée selon une hiérarchie variable. À un moment donné, sur un projet donné, l'observateur extérieur pourra déduire une certaine hiérarchie entre les personnes, mais que l'instant ou le projet changent, et la hiérarchie aura bougé. Une personne peut en effet être à la fois dans l'équipe et chef d'équipe d'une autre personne, selon le projet ou les compétences en jeu. Il reste quand-même quelques points stables, avec notamment le rôle du directeur général qui oriente la stratégie, et la structure administrative (gestion des congés par exemple).

Chaque responsable hiérarchique a plusieurs attributions :

- organiser le travail, confier des tâches,
- demander des compte-rendus, des documents et avoir accès à toute la production,
- gérer les calendriers, autoriser les congés,
- fournir *reporting* et analyse des résultats au directeur général.

Makina Corpus fonctionne en mode projet ; la hiérarchie n'est pas permanente mais dépend du projet en cours :

- le responsable commercial pour les activités commerciales,
- le directeur de mission dans les missions de conseil qui le nécessitent,
- le responsable de la production sur les projets de développement ou pendant les intermissions

Afin de maintenir un esprit communautaire au sein de Makina Corpus, il est important que la hiérarchie soit la plus légère et la plus souple possible. Cette organisation qui évite les pesanteurs d'une hiérarchie, donne de l'air à chacun en lui permettant de s'essayer à divers rôles. Cette possibilité est un des attraits de Makina Corpus pour les prétendants à l'embauche.

Responsable de la production

Son rôle est de :

- qualifier la demande avec le commercial en charge du prospect (accord sur objet de la prestation, étendue, livraisons, délais, durée, ressources),
- désigner les développeurs les plus à même d'effectuer les travaux,
- définir et faire respecter le calendrier et s'assurer du respect des durée de développement par les équipes,
- garantir la qualité du travail et appuyer les développeurs,
- valider les orientations techniques,
- former et informer l'équipe commerciale sur tous les outils, les produits, les développements et les compétences techniques développées par Makina Corpus.

Le rôle comprend également une responsabilité hiérarchique sur les équipes pilotées.

Le responsable de la production doit dialoguer avec l'ensemble de l'équipe de production et accepter certains des choix techniques pris au niveau global de la société (par exemple Développement des compétences Zope). Il doit également être l'interlocuteur technique privilégié des commerciaux qui viendront lui demander des estimations de temps, des renseignements techniques...

Sur des projets de développement particulièrement longs ou lourds, le responsable de la production nomme un chef de projet au sein des développeurs. Ce dernier devient alors l'interlocuteur privilégié du commercial.

Les missions longues (notamment les régies) sont gérées par des directeurs de missions.

Les directeurs de mission

Un directeur de mission est désigné pour toute mission de conseil et intervient à trois niveaux : avant, pendant et après la mission. Il est le médiateur entre le client et le consultant, entre le consultant et Makina Corpus ; il est leur contact privilégié. Il travaille de concert avec le commercial, le consultant, l'administration, et le client.

Il intervient à toutes les étapes de la réalisation de la mission :

- il négocie, avec le commercial, les détails de mission auprès du client,
- il prépare le consultant chargé de la mission,
- il suit la réalisation de la mission et veille à sa bonne exécution,
- il s'assure en permanence de la satisfaction du client et garde le contact avec lui.

Le directeur de mission est responsable auprès de la direction générale du bon déroulement de la mission et de la qualité de la prestation. Il rend compte de la mission lors de réunions bimensuelles auprès de la direction générale. Il est le responsable hiérarchique du consultant tout au long de la mission et jusqu'à la période de capitalisation.

La qualité principale d'un tel perosnnage est la compétence dans la gestion de la relation avec le client. Des compétences sur le sujet sont appréciées, mais pas nécessaires. Noter par ailleurs que des directeurs de mission peuvent très bien appartenir à d'autres sociétés avec lesquelles Makina Corpus travaille, si les projets communs et les profils disponibles le rendent souhaitables. Dans ce cas, l'intervenant extérieur sera formé au rôle de directeur de mission à la Makina Corpus s'il ne l'a jamais endossé.

Responsable commercial

Responsabilités :

- gérer les prescripteurs,
- valider les axes de prospection des commerciaux et leur évolution,
- gérer les priorités des commerciaux,
- répartir les flux commerciaux entrant entre les consultants (sauf quand il s'agit de prospection directe),
- s'assurer de la satisfaction du client.

Le responsable des activités commerces pilote également actuellement de près deux consultants ayant des réseaux et/ou carnets d'adresses à exploiter largement. Cette relation particulière, ponctuelle, pourra être activée à nouveau selon le profil de tel ou tel nouveau/elle recruté/e. Ceci constitue un petit noyau commercial stable, autour duquel gravitent tous les autres consultants qui peuvent en fonction des besoins apporter leurs ressources.

Ce noyau commercial stable (dans une hiérarchie mouvante) est aussi l'assurance que l'ensemble des relations commerciales de Makina Corpus avec l'extérieur est filtré. Pour leur tranquilité, aucun développeur ne doit pouvoir être contacté en direct par un nouveau client.

Chacun est vivement encouragé à participer aux activités commerciales en réfléchissant à des pistes possibles et en exploitant ses propres connaissances. Il est nécessaire d'associer le responsable commercial à cette démarche afin de maintenir une forte cohérence des actions (un même client ne doit pas être contacté par trois interlocuteurs différents de Makina).

En résumé, si un nouveau client prend contact avec MC, il doit être dirigé vers l'équipe commerciale. Si un autre salarié est en contact par ses propres moyens avec un client potentiel, il peut le gérer.

Responsable Marketing

Son rôle :

– mise en œuvre du marketing de la société,
– réalisation des différentes actions opérationnelles,
– présentation continue des actions et résultats au directeur général.

Les activités marketing se font au service du commercial et de la société et consistent plus généralement à intervenir de façon globale au nom de l'ensemble de la société et d'en coordonner les activités . Le responsable du marketing fournit des outils, des actions à destination de clients et de prospects connus ou inconnus pour le compte de l'ensemble de la société. Il dépend hiérarchiquement du directeur général, mais celui-ci étant peu disponible, est actuellement temporairement rattaché au responsable commercial.

5.6.2 Qui sont-ils ?

Il n'existait pas de communauté qui précisément ait donné lieu à la constitution de Makina Corpus. Le premier employé de Makina travaillait auparavant avec le directeur général dans une autre société, et avait construit un profil proche de ce qu'est le consultant Makina Corpus aujourd'hui. Ils ont alors réfléchi à la construction d'un structure nouvelle, organisée autour du développement, du conseil et des services. Assez rapidement, le profil particulier de Mose a été attiré, et par capillarité[18], une bonne moitié des makinistes proviennent de Claranet, comme et à la suite de Mose.

Ce groupe de personnes constitue certainement le noyau structurant de Makina Corpus, qui lui confère une partie de son esprit et de son ambiance de travail. Ils n'avaient que peu de temps chez leur employeur précédent pour se consacrer au Libre, et on osera dire que leur arrivée chez Makina les a libéré sur ce plan. Habitués à travailler ensemble[19], ils s'apprécient et apprécient pouvoir continuer à opérer de concert.

Noter que ce groupe de personnes[20] était constitué aussi bien de développeurs que d'utilisateurs ou d'utilisateurs / développeurs, ayant eu des parcours universitaires différents. On retrouve dans ce groupe les divers profils (expérimentés et avertis ou non, techniciens ou non) que l'on trouvera chez les clients. Les Makinistes observent d'ailleurs que la collaboration autour du développement libre se fait rarement avec des groupes, mais plutôt avec des sommes d'individus très variés qui ne sont pas souvent de la même commmunauté.

L'organisation en géométrie variable de la société découle de l'organisation informelle et non consolidée au départ de ce noyau. Un noyau constitué de plusieurs individus se rapprochant tantôt de l'un tantôt de l'autre en fonction des sujets, le credo commun étant le Libre[21].

De tels noyaux sont fragiles : il peut suffir d'un seul humain charismatique pour être le noyau d'une communauté qui se développe, mais au-delà d'une dizaine de personne le noyau peut facilement éclater. La plupart des grandes réussites du Libre ne sont maintenues que par un groupe très restreint de personnes, voire un seul, qui parfois permute. C'est ce type d'observations qui a conduit à l'organisation actuelle de Makina

[18]La cooptation est une des méthodes de recrutement.
[19]et à utiliser les outils de travail collaboratif développés par eux-même.
[20]au sein duquel on peut même identifi er deux pré-communautés géographiques, Marseille et le Bretagne
[21]pas seulement circonscrit aux logiciels mais comme une alternative de pensée plus générale

Corpus, qui doit à la fois gérer un bouillonnement interne[22] et une cohérence externe.

La société a beaucoup évolué en un an. D'un groupe sans hiérarchie, on passe à une structure plus organisée avec des flux préétablis (comment traiter les demandes, rentrer en relation avec les clients, suivre les calendriers...). Structure plus organisée certes, mais également plus importante, qui ne permet plus de réunions hebdomadaires à échelle humaine (surtout quand certains employés ne peuvent être présents car en régie chez un client). Il faut donc inventer et s'approprier de nouveaux moyens pour continuer à organiser la structure en faisant participer tout le monde. Ce que Makina devrait pouvoir faire sans trop de difficultés puisqu'elle préconise les outils logiciels qui permettent ce type d'organisations d'une part, et a été constituée sur un noyau de personnalités qui s'apprécient.

Makina laisse-t-elle une autonomie à ses employés pour continuer à développer tel ou tel logiciel libre ? Un élément de la réponse est que les outils Makina Corpus sont des logiciels libres. Une grande partie du temps est passé à les développer. Lorsqu'un salarié souhaite se consacrer à un logiciel libre non encore utilisé chez Makina Corpus, cette dernière essaye de l'intégrer à son portefeuille d'outils standards et d'en faire une des forces de la société. Comme le dit l'un des consultants : «Pour l'instant les employés ont une liberté proportionnelle à leur culot temperé par leur sens des responsabilités et une modération quotidienne de pair à pair.». Il n'existe pas de clause spéciale dans le contrat de travail à ce sujet.

Il est clair que l'«esprit du Libre» déteint sur l'ensemble des activités de la société. Dans la gestion quotidienne, les habitudes de collaboration, le soucis de transparence, la préoccupation de la redondance, l'envie du partage, ce sont autant de choses qui du technique déteignent sur l'humain.

5.6.3 Recrutement

Le recrutement s'effectue par cooptation (ce qui renforce le noyau de départ), via les sites internet (en particulier ceux des communautés Libre) et plus classiquement via l'APEC. Il faut compter également avec de très nombreuses demandes d'embauche spontanées. Choisir tel ou tel profil n'est pas chose aisée tant la construction d'un noyau de départ cohérent sera la source de la réussite de cette société dans ses premières années.

À la question «comment faire pour attirer et retenir des développeurs qui nagent entre deux eaux, ni dans la communauté ni dans des entreprises ?», Makina répond :

«Les techniciens spécialisés, dans notre métier, travaillent plus pour des projets que pour des sociétés[23] , le cycle de l'emploi est précaire et pourtant peu dans l'informatique ont peur de se retrouver au chômage. La gestion d'un *turn over* ne doit pas être un problème, et la communauté du Libre, en fournissant quelques repères de valeur communs, permet de passer outre l'éclectisme. De toute façon, un développeur n'est précieux que quand il est motivé, tout expert qu'il soit, et l'émulation autour du libre a sans doute aussi pour but d'y contribuer.»

5.7 Interactions

Makina Corpus n'est pas encore membre d'associations particulières mais cela devrait rapidement changer (certains consultants y sont cependant déjà à titre personnel). Un engagement dans le logiciel libre se

[22]«des coups de *speed* initiés par des individus suivis de périodes d'isolement (au sens de concentration dans le travail)»

[23]et pour cela ne craignent pas la mobilité de l'emploi et usent des facilités du portage salarial.

traduit souvent par une position militante pour se défendre face à des lobbies fort bien organisés.

Les salariés dans leur ensemble participent à de nombreux réseaux (ils en sont parfois les fondateurs et les animateurs). De leurs vies antérieures à Makina Corpus, ils gardent des contacts personnels avec de nombreux décideurs d'associations autour du libre —alors même que les membres de ces associations ne sont pas encore largement conscientes de l'existente de Makina Corpus . En revanche, les autres sociétés de logiciel libre en ont rapidement pris conscience et ont établis des contacts, les secteurs d'activité de prédilection de Makina Corpus étant particulièrement motivants pour les équipes de développeurs..

Makina a commencé depuis peu à travailler avec d'autres sociétés orientées Libre, principalement depuis qu'elle estime avec des prestations d'envergure et originales à proposer. Le tout est de reconnaître les compétences et particularismes de chacunes des structures. Le service en ingénierie informatique couvre un très vaste champ de compétences, et être pertinent sur toutes ne semble ni possible et ni souhaitable pour une structure unique. C'est pourquoi, pour founir une réponse complète à une demande client couvrant un large domaine, le partenariat co-traitance sous-traitance semble être une voie viable.

Ici, l'esprit du Libre est clairement un avantage, car il favorise une ouverture a priori du fait qu'il n'y a pas à se soucier de protéger ses sources quand on collabore avec un 'concurrent'. Makina se joint principalement à d'autres SS2L pour répondre a des appels d'offre ambitieux[24] . Parmi les contacts de Makina se trouvent Idealx, Nuxeo, Easter-eggs, Lolix, et d'autres. C'est aussi la taille (réduite) de ces différentes structures qui les poussent naturellement à répondre ensemble à des projets d'envergure.

Pour terminer cette monographie, nous avons demandé aux consultants de Makina Corpus s'ils avaient le sentiment qu'il y avait bijection entre «faire du business avec du Libre» et «développer du Libre», ou qu'au contraire on voit (de plus en plus ?) de parasites/tage.

Philippe : «Pour certain il y a une ambiguïté de traduction *free software* logiciel libre et logiciel gratuit, ce qui peut amemer a un sentiment de bijection, genre je fait du commerce avec un truc gratuit [ce n'est pas correct], etc. Pour moi il n'y a pas d'ambiguïté dans la traduction. *Free software* donne logiciel libre, c'est-à-dire un logiciel dont tu peux récupérer les sources et l'installer, libre à toi de le personnaliser ou pas. Maintenant, s'il se trouve que tu n'as pas les compétences, le temps, l'envie de faire ça et que tu décide de payer quelqu'un pour le faire, je ne vois rien de scandaleux à ce que tu le fasses, et donc que des gens te proposent ce service. En fait, je rapproche ça de la mécanique sur véhicule ancien : j'ai eu quelques vieilles moto et j'aurais pu payer quelqu'un pour les remettre en état de marche, ça ne m'aurait pas choqué, même en sachant que comme pour le logiciel libre les 'sources' de l'assemblage mécanique de pièces composant cette moto sont en libre accès et que donc je le paye pour un truc que j'aurais eu les moyens de réaliser seul.»

Mose : «Tel que c'est formulé, je dirai qu'on peut développer du Libre et ne pas en vivre, mais que faire [de l'argent] avec le Libre n'impose pas forcément de contribuer, mais plutôt d'exploiter la capitalisation de la communauté. Il y a la un deséquilibre notable, mais qui ne me choque pas plus que d'autres qu'au quotidien je remarque. En tout cas mon sentiment est que s'il s'agit de gagner de l'argent avec du Libre, un bon moyen de s'y prendre c'est d'en faire au moins un peu [(de Libre)].»

[24]On a dit ailleurs que Makina ne répondait qu'à des appels d'offres émanant de personnes connues ; ici, le demandeur peut être connu et recommandé par une autre SS2L.

Deuxième partie

Analyse économique des stratégies de valorisation de la production de logiciel.

Les sociétés de services en logiciels libres. L'émergence d'un système de production alternatif au sein de l'industrie du logiciel ?

MARIE CORIS,
Université de Bordeaux Montesquieu, coris@montesquieu.u-bordeaux.fr.

Si l'intérêt que l'on peut porter au mouvement Libre s'est à ce point développé ces dernières années, il est lié au mouvement de diffusion de Linux. L'émergence commerciale de ce système d'exploitation a permis que se crée un marché pour l'ensemble des logiciels libres qui ne sont plus cantonnés à des niches très spécifiques mais qui peuvent désormais faire l'objet d'une problématique d'ensemble dans la mise en œuvre d'un système d'information. Linux a engendré cet essor et cette reconnaissance marchande des logiciels libres parce qu'il est un système d'exploitation, c'est à dire l'élément central de toute architecture informatique en ce qu'il permet la communication entre les logiciels, entre les logiciels et la machine et entre l'homme, la machine et les logiciels (Dréan [1996]). Cependant, le phénomène lié au relatif succès commercial de Linux, qui détiendrait, d'après une étude d'IDC, environ 25 % des parts sur le marché des serveurs[1], ne saurait se restreindre à la simple, quoique primordiale, question de compétition technologique entre standards logiciels puisque les logiques de production elles-mêmes s'opposent. Si les logiciels commerciaux ou propriétaires sont le résultat de l'exercice des droits de propriété intellectuelle permettant au détenteur du standard de marché l'exercice d'un pouvoir fort de monopole pouvant conduire au verrouillage du marché (Shapiro et Varian [2000]), les logiciels libres sont issus d'une tradition scientifique d'ouverture et de mise à disposition du code source. Le logiciel libre est l'apanage des communautés qui, aidées de l'outil de coopération qu'est le réseau Internet, développent des logiciels en dehors de la sphère marchande. Parce que dans le cas de Linux, le système d'exploitation était d'une très grande fiabilité technique et parce qu'il permettait d'envisager une alternative à Windows, des sociétés de distribution se sont créées afin de le commercialiser et nombre des acteurs traditionnels de l'informatique l'ont intégré dans leur offre commerciale. Car si la fiabilité technique dont jouissent Linux et les logiciels libres en général, est le fruit du travail des communautés de développeurs, souvent bénévoles, la reconnaissance marchande du système d'exploitation s'explique aussi par la structuration d'une offre commerciale, matérielle et logicielle, permise par la conjonction de deux éléments. Il faut tenir compte, d'une part, de la création de sociétés de distribution comme RedHat, Mandrake ou SuSE ayant pour vocation de faire de Linux, alors uniquement accessible aux experts, un produit commercial, convivial dans son installation et son utilisation. D'autre part, l'existence d'un marché pour les logiciels libres doit aussi beaucoup au ralliement de nombre d'acteurs traditionnels tels IBM, Sun ou HP, venant structurer l'offre de produits complémentaires (Maume [2002]) par la portabilité de progiciels commerciaux sous Linux (Oracle) ainsi que par celle des composants matériels. Ainsi la niche de marché peut se créer, et Linux devenir une véritable alternative technologique aux solutions commerciales déjà éprouvées telles celles de Microsoft.

Au-delà de la création d'un segment de marché orienté «Linux», des sociétés de services spécialisées dans les logiciels libres (SSLL) sont nées du phénomène afin de proposer aux entreprises les mêmes services que les SSII (Sociétés de Services et d'Ingénierie Informatique) traditionnelles[2], mais dans la sphère des logiciels libres. Le but est de faire du développement logiciel libre une stratégie commerciale, un nouveau système de production au sein de l'industrie du logiciel, non plus cantonné aux communautés de développeurs bénévoles, mais s'appuyant sur les développements réalisés par ces communautés. Il ne s'agit alors plus seulement de la possible substitution de Linux à Windows ou à quelque autre système d'exploitation, mais de l'émergence d'une alternative plus vaste au sein de l'industrie du logiciel, basée sur un nouveau système de production tentant de supplanter le développement logiciel propriétaire par le Libre. En d'autres termes, le succès commercial de Linux ne présage-t-il pas de l'émergence d'un système de production alternatif au sein de l'industrie du logiciel elle-même ?

Actuellement, les SSLL tentent de se dégager un avantage concurrentiel au sein d'une industrie historiquement occupée par le couple SSII-Editeurs de progiciels. Ce n'est plus avec les éditeurs qu'elles doivent mettre en œuvre leur système de production, mais avec les communautés de développeurs. En vue de caractériser

[1] http://www.idc.com

[2] À l'échelle de la France, ces SSLL se sont crées depuis 1997 et tentent de concurrencer le traditionnel couple Éditeurs-SSII sur le marché professionnel.

ce système et de prolonger les monographies d'entreprises proposées dans les articles précédents, qui étaient des exemples de la construction de tels modèles, une première synthèse est ici proposée, basée sur une étude de terrain d'une dizaine des principales SSLL françaises menée entre janvier 2001 et février 2002 sous forme d'entretiens semi-directifs auprès des responsables et fondateurs de ces sociétés et conduite en s'appuyant sur une approche en terme de système de production ou de modèle productif, développée par Boyer et Freyssenet [1995], [2000]. Il convient alors de définir le système de production à partir des tentatives de résolution apportées par les SSLL aux deux incertitudes fondamentales auxquelles toute entreprise est soumise : celle du marché et celle du travail. Rien ne garantit à l'entreprise que ses investissements en capital seront rentables, c'est à dire qu'elle trouvera des débouchés pour les biens et services produits ; de même, elle ne saurait *a priori* être assurée d'obtenir de ses salariés la production voulue selon les conditions souhaitées. S'interroger sur le système de production que les SSLL mettent en œuvre conduit à mener deux investigations : 1) l'identification, compte tenu du contexte concurrentiel d'émergence, de l'avantage concurrentiel que les SSLL peuvent tenter de dégager face à la domination historique du couple SSII-Éditeurs, ce qui revient à étudier les segments de marché visés et les services effectivement offerts ; 2) l'analyse des spécificités organisationnelles des SSLL, ce qui amène à interroger deux aspects de l'organisation de la production de services, la relation entre communautés de développeurs et SSLL d'une part, et la mise en œuvre de la gestion de la relation au sein des SSLL, d'autre part.

6.1 Gérer l'incertitude de marché : quel avantage concurrentiel pour les SSLL ?

Toute entreprise, lors de sa création ou de la production de nouveaux produits ou services, est soumise à l'incertitude du marché quant à la vente effective de ceux-ci. Pour réduire cette incertitude, liée à l'environnement productif au sein duquel les entreprises émergent, à savoir ici l'état de l'industrie du logiciel (1), l'entreprise doit tenter de se dégager un avantage concurrentiel durable face à ses concurrentes, c'est-à-dire cerner l'étendue demande solvable en fonction des priorités des acheteurs potentiels (Boyer et Freyssenet [2000]). La recherche d'un avantage concurrentiel durable face au modèle «traditionnel» des Éditeurs-SSII amène les SSLL à cibler certains segments de marchés *a priori* profitables compte tenu de l'environnement concurrentiel identifié, en vue d'y proposer une gamme de services selon une stratégie de profit misant, ici, sur la très forte personnalisation des services offerts (2).

6.1.1 L'inscription des SSLL dans leur environnement concurrentiel.

Linux et les principaux logiciels libres constituent des alternatives technologiques aux standards propriétaires établis. Bien que ces logiciels soient des innovations de produits, il n'y a pas à proprement parler de rupture technologique dans le sens où de nouvelles demandes, de nouveaux besoins ne vont pas se structurer et s'exprimer sur des marchés émergents. En effet, d'autres acteurs sont déjà présents là où apparaissent ces logiciels, comme Microsoft dominant le marché des systèmes d'exploitation pour serveurs PC avec Windows NT lors de l'émergence de Linux. On pense aussi aux nombreuses applications qui se sont développées ; si Oracle est portée sous Linux, d'autres systèmes de gestion de bases de données (PostgrepSQL, MySQL) libres essaient de se positionner face au standard propriétaire. Cependant, Linux représente non seulement l'emblème des logiciels libres, mais il est aussi, en tant que système d'exploitation, l'élément clef de la diffusion de l'ensemble des logiciels libres. L'activité des SSLL, et surtout sa pérennité, sont donc liées à l'état de diffusion de Linux ; la compétition technologique entre Windows (standard du marché) et Linux apparaît alors de prime importance dans l'étude de l'environnement concurrentiel des SSLL. Son analyse nous montre qu'il se caractérise par un marché en apparence verrouillé et la création d'un segment de marché centré sur Linux.

Un marché en apparence verrouillé.

La nature même du logiciel en tant que bien d'information permet l'obtention d'un monopole temporaire pour un éditeur de progiciel commercial (Shapiro et Varian [2000]) ; l'exploitation des rendements croissants d'adoption (RCA[3]) peut, quant à elle, engendrer le verrouillage de la base installée d'utilisateurs. Lorsque les RCA concernent un élément central comme l'est le système d'exploitation, le verrouillage peut concerner toute ou partie de l'industrie liée. Foray [1990] note que le degré du verrouillage va dépendre de l'importance des rendements croissants d'adoption. En reprenant le cadre d'analyse proposé par Arthur [1989a] et développé par Foray [1989], [1990], on peut montrer que, dans le cas des systèmes d'exploitation, les cinq sources de rendements croissants d'adoption existent :

1. la puissance détenue par Microsoft sur les marchés des systèmes d'exploitation pour ordinateurs personnels ou pour serveurs PC s'explique partiellement par les économies d'échelle de production qui permettent l'établissement d'un monopole temporaire (importance des coûts fixes et faiblesse des coûts marginaux) ;
2. les économies d'échelle de production sont renforcées par les économies d'échelle liées à la demande, dues à la présence d'externalités de réseau : plus une technologie est adoptée, plus son utilité augmente pour l'usager, en raison directe de l'accroissement de la base installée des utilisateurs (partage de fichiers, par exemple) ;
3. afin d'établir le verrouillage de la base installée d'utilisateurs et d'augmenter les externalités de réseau, un éditeur de progiciel générique peut exploiter les coûts de changement qui s'appréhendent comme l'ensemble des coûts directs (acquisition de nouveaux logiciels, par exemple) et des coûts induits, comme l'apprentissage, qui caractérise bien l'enfermement de la base installée sur un système d'exploitation et les logiciels applicatifs liés. Parce que l'utilisateur lambda est habitué à se servir d'un progiciel et que l'éditeur garantit une certaine rétro-compatibilité entre les versions successives du progiciel en question, il n'aura, *a priori*, aucun intérêt à changer de système en raison des coûts de changement induits ;
4. dans le cas des biens d'information, les coûts de changement seront d'autant plus importants que le niveau d'interdépendance avec des produits complémentaires (les logiciels applicatifs, les composants matériels) est élevé. Il s'agit ici de la notion d'interrelations technologiques. Plus nombreuses sont les technologies liées venant structurer l'environnement d'une technologie, plus celle-ci sera attractive ;
5. les rendements croissants d'information, c'est-à-dire le fait que plus une technologie est diffusée, plus elle est connue, permettent aussi d'expliquer l'inertie au changement des utilisateurs, particuliers comme entreprises, puisque plus la technologie est adoptée et moins l'aversion au risque constituera un facteur de blocage à sa diffusion, ce qui revient à se rallier à la solution dominante parce qu'elle est dominante (c'est la publicité d'IBM : «personne ne s'est jamais fait viré pour avoir acheté de l' IBM»).

Ainsi, il ne s'agit pas d'un monopole temporaire mais d'un véritable verrouillage, non seulement de la base installée des utilisateurs, mais aussi de toute une part de l'industrie qu'une entreprise comme Microsoft peut plus ou moins contrôler du fait de son monopole sur le système d'exploitation. Remettre en cause un tel verrouillage par un système d'exploitation issu de la même logique productive ne ferait que le déplacer sur une autre solution propriétaire.

Par ailleurs, et c'est presque un sixième point, le cadre de propriété intellectuelle, relevant de la volonté des institutions nationales ou supra-nationales, revêt ici une importance de taille qui permet d'insérer les SSLL, comme l'industrie du logiciel, dans leur cadre macro-économique et de mettre l'accent sur une autre source, plus institutionnelle cette fois, favorisant le verrouillage. La GPL (GNU General Public License)

[3]Le processus de diffusion d'une technologie est auto-renforçant, dans le sens où plus elle est adoptée, plus elle sera adoptée. Ce processus de diffusion auto-renforçant est formalisé par Arthur [1989a] grâce à la notion de Rendements Croissants d'Adoption.

protégeant les logiciels libres pose des questions concernant sa validité au regard du droit français (Clément-Fontaine [1999]). Pour que Linux et les logiciels libres se diffusent, il est nécessaire que les institutions publiques les reconnaissent, ce qui peut passer par la simple adoption et utilisation de ces logiciels, car en les utilisant, elles leur offrent une reconnaissance légale. Plus préoccupante que la simple question de la validité totale ou soumise à restrictions de la GPL[4] reste la question de la possible brevetabilité des logiciels (Zimmermann [1995b], [1999]) et de son impact sur les logiciels libres[5].

Ainsi Linux émerge sur un marché en apparence verrouillé par Windows. Une analyse plus fine de l'état du verrouillage en fonction des segments de marché (celui des particuliers avec les systèmes d'exploitation pour postes clients ou ordinateurs personnels et celui des professionnels avec les systèmes d'exploitation pour serveurs) montre que le verrouillage apparaît moins étendu, donc réversible, sur le marché des serveurs (utilisateurs avertis), ce qui permet d'y envisager la création d'un segment de marché «Linux».

La création du segment de marché centré sur Linux.

Si l'on introduit une distinction des types d'utilisateurs selon un critère simple (devant admettre une certaine graduation) fonction de la maîtrise technique de l'outil informatique, on peut alors distinguer deux grandes classes parmi les utilisateurs de l'informatique : celle des non-informaticiens (utilisateurs non avertis) et celle des professionnels de l'informatique (utilisateurs avertis). Parce qu'il ne bénéficie en général d'aucune formation particulière hormis celle offerte à l'école et concernant en général les standards du marché, l'utilisateur lambda apparaît plus adverse au risque que l'utilisateur formé pour qui, compte tenu de sa maîtrise de l'outil, la migration d'un système à un autre sera plus aisée. Par ailleurs, il faut considérer la divergence d'utilisation faite de l'informatique par les particuliers et par les professionnels. L'utilisateur domestique ne voit pas dans l'informatique un outil de travail pour applications plus ou moins critiques, mais plus un outil de divertissement ou de gestion domestique. De même, il ne recherche pas particulièrement la fiabilité mais la convivialité des outils logiciels. Il s'agit alors d'un marché de masse, particulièrement profitable pour les éditeurs de progiciels commerciaux et conviviaux. D'un autre côté se situe la classe des professionnels, ou du moins celle des utilisateurs avertis, où la situation de verrouillage apparaît moins irréversible, compte tenu des besoins des utilisateurs et des avantages que peuvent alors fournir les logiciels libres de type Linux. Par exemple, la formation dont bénéficient les professionnels réduit les effets d'apprentissage liés à l'utilisation d'un système d'exploitation particulier, ce qui réduit les coûts de changement induits.

De même que l'on a introduit une différence de degré de verrouillage possible en fonction du type d'utilisateurs concernés, il faut poursuivre l'analyse en distinguant deux types de segments de marché pour les systèmes d'exploitation : celui des postes clients et ordinateurs personnels et celui des serveurs PC[6]. Les PC ont été introduits sur le marché au cours des années 70 et étaient dédiés à un usage personnel. Le PC n'était ainsi pas conçu pour remplir la fonction de serveur alors assurée par les stations de travail Unix. Avec l'usage grandissant de la mise en réseau des ordinateurs (réseaux internes et externes à l'entreprise), et afin de répondre au besoin d'homogénéisation des parcs informatiques des entreprises, le PC a été, au début des années 90, modifié techniquement afin de devenir serveur (d'où l'utilisation du terme de serveur-PC). Parce que Microsoft détenait déjà un quasi-monopole sur le marché des systèmes d'exploitation pour ordinateurs personnels, la version pour serveurs (NT) de Windows est devenue dominante sur ce nouveau segment de marché. Cependant, deux éléments relatifs à l'émergence de Linux sont à prendre en considération, relativisant l'ampleur du monopole et la possibilité du verrouillage sur ce segment de marché. D'une part,

[4]La GPL exclut toute responsabilité de la part des auteurs du logiciel. Les sociétés de distributions surmontent ce problème en introduisant une clause de garantie dans le contrat de licence.

[5]Pour certains auteurs, le brevet pourrait contrarier l'avenir du libre et par conséquent celui des SSLL (voir, par exemple Faucon et Smets-Solanes [1999]).

[6]L'analyse menée ici doit beaucoup au travail réalisé par Jullien [1999].

le segment de marché des serveurs PC est un nouveau marché au moment ou Linux entre dans la vision des décideurs informatiques, ce qui signifie que si quasi-monopole il y a, la situation de verrouillage est encore très largement réversible. D'autre part, si les serveurs-PC, parce qu'en tant que machines ils atteignent rapidement les performances des stations de travail, se diffusent rapidement et s'ils sont financièrement plus abordables, les entreprises souhaitent que le système d'exploitation pour ces serveurs-PC recouvre les mêmes caractéristiques techniques qu'Unix. Et Linux n'est autre qu'un «Unix pour PC». Ces deux arguments pris conjointement peuvent expliquer pourquoi la diffusion de Linux en tant que système d'exploitation pour serveurs-PC se réalise plus rapidement que sur le marché des postes clients.

Mais le fait que le verrouillage apparaisse moins irréversible sur le marché des professionnels, et particulièrement sur le segment des serveurs, ne suffit pas à expliquer la diffusion de Linux que l'on y observe. En effet, quel intérêt pourrait-il y avoir pour un utilisateur à envisager la migration vers une solution strictement équivalente toutes choses égales par ailleurs ? Il faut que la nouvelle technologie présente un avantage aux yeux de l'utilisateur afin qu'il soit encouragé à l'adopter. Dans le cas de Linux, la très grande fiabilité technique dont jouit le système d'exploitation (Horn [2000b]) fut un élément déterminant pour son adoption. Les logiciels libres (Apache ou Linux constituent à ce propos des exemples patents), parce qu'ils sont réputés pour leurs qualités techniques et leur fiabilité lors de leur mise en œuvre dans des applications critiques, répondent alors aux besoins d'utilisateurs professionnels dans certains domaines où la fiabilité apparaît comme un élément critique (système embarqué, temps réel, ...)

Cette analyse, menée au niveau théorique, permet de cerner dans quel contexte ont émergé les SSLL dès 1997, dont l'expérience vient confirmer l'analyse. Cependant, une étude en terme de compétition technologique entre deux produits ne saurait être suffisante, et la diffusion de Linux n'aurait su se faire sans la contribution des sociétés de distribution ainsi que sans le ralliement de certains grands noms de l'informatique. L'ensemble des responsables des SSLL interrogés s'accorde à dire que le Libre manquait à ses débuts d'une aura professionnelle auprès des clients potentiels. D'autre part, le système d'exploitation étant l'élément central dans l'architecture du système d'information, les interrelations technologiques apparaissent de prime importance dans la possibilité de sortir du verrouillage. En assurant la compatibilité matérielle et logicielle, des acteurs comme Sun, HP, IBM, Oracle, réduisent sensiblement les coûts de changement directs et réduisent[7] la source de RCA relative aux interrelations technologiques. La compatibilité des applicatifs, permettant l'échange de fichiers ou de données que les progiciels soient utilisés sous Windows ou sous Linux, vient réduire l'effet associé aux externalités de réseau. Si l'offre de progiciels compatibles n'a cessé de s'accroître, elle était minime lors de la création des premières SSLL, le marché étant alors cantonné à un nombre très restreint d'applications (les serveurs web par exemple). Aujourd'hui, de plus en plus d'applicatifs sont disponibles, venant réduire les coûts de changement en terme de champs d'application, bien que des domaines, comme la comptabilité, restent très mal pourvus. Le ralliement d'un acteur comme IBM, offre à Linux et aux SSLL la crédibilité dont ils avaient besoin. Qu'IBM veuille ici recouvrer des parts de marché et favoriser son offre matérielle, les responsables reconnaissent le rôle indéniable et décisif de cet acteur, par les investissements réalisés pour Linux[8], son intégration dans l'offre de la firme (Lin [2002a]) et par la communication agressive sur Linux (publicité diffusée sur les chaînes hertziennes).

Ainsi se crée un segment de marché autour du système d'exploitation Linux, et parce que leur activité dépend de la diffusion du système d'exploitation, les SSLL vont y émerger pour tenter de se positionner face aux SSII. De même que le verrouillage est moins important sur le marché des professionnels, la diffusion de Linux y sera plus rapide et l'opportunité pour les SSLL d'y être profitable, réelle.

[7] Il faut relativiser le propos. Les architectures logicielles sont très complexes et nombreux sont les logiciels dits «applications métiers» qui ne sont pas «portés» sous Linux ou qui ne trouvent pas leur équivalent dans la sphère des logiciels libres.

[8] Voir à ce propos le site d'IBM : http://www-1.ibm.com/linux/.

6.1.2 Le positionnement des SSLL : quel avantage concurrentiel, sur quels segments de marché, pour quelle offre de services ?

Avec la création d'un segment de marché «Linux», le système d'exploitation entre dans le champ de vision des entreprises qui le perçoivent alors comme un produit parmi d'autres. Ceci ne saurait témoigner de l'émergence d'un nouveau système de production mais signifie simplement l'avènement d'une technologie alternative au standard en place, en l'occurrence à Windows. Les distributeurs peuvent alors jouer le rôle des éditeurs pour Linux et les SSII, dont l'une des fonctions est d'intégrer et d'adapter aux besoins précis des utilisateurs les progiciels fournis par les éditeurs, peuvent l'incorporer dans leur offre. Parce qu'elles concentrent leur activité dans la sphère des logiciels libres, les SSLL ouvrent, quant à elles, la voie à un système de production alternatif au sein de l'industrie du logiciel, système de production que nous allons étudier en envisageant d'une part l'avantage concurrentiel qu'elles peuvent se dégager et d'autre part la politique-produit des SSLL, c'est à dire leur métier, l'ensemble des services effectivement offerts.

L'avantage concurrentiel par les coûts et la fiabilité.

Les SSLL émergent sur un marché où les clients visés dépendent d'autres prestataires, les éditeurs, pour la fourniture des progiciels applicatifs et des logiciels systèmes, et les SSII pour le l'intégration, l'adaptation et la personnalisation de ces logiciels ou pour le développement de logiciels spécifiques. En arrivant sur un marché historiquement dominé par le couple Éditeurs-SSII, les SSLL vont devoir se doter d'un avantage concurrentiel. Cela renvoie à la préoccupation de déterminer sur quoi elles peuvent jouer pour attirer les clients et sur quels segments de marché elles doivent se positionner.

Tout comme les distributeurs de Linux, les SSLL vont jouer de l'avantage de la fiabilité, intrinsèque au logiciel libre. Car si Linux jouit d'une telle popularité, ce n'est pas parce qu'il incarne une certaine éthique informatique, mais parce qu'il bénéficie d'un écho extrêmement favorable concernant sa fiabilité technique. Cependant, nous avons identifié un certain état de verrouillage qui débute lorsque, justement, une technologie qui serait préférée, toutes choses égales par ailleurs, ne sera pas choisie compte tenu de l'état de diffusion d'une autre technologie, alors adoptée. Ceci se traduit par le retard d'adoption de la technologie en question, dû à l'excès d'inertie alors en œuvre sur le marché, où chacun attend que les autres se décident à adopter une nouvelle technologie avant de le faire[9]. Il faut que le logiciel se diffuse pour qu'il soit adopté ; le choix de la clientèle à cibler est alors de prime importance. Les responsables des SSLL interrogés dont la société était présente sur le marché dès 1997-1998 reconnaissent avoir, dans un premier temps, visé le marché des PME/PMI de type industriel, pour finalement se concentrer sur les grands comptes et les administrations. Ici, l'étude de terrain vient valider l'hypothèse théorique que nous pouvons formuler *a priori* : la nécessité de jouer sur la qualité technique afin d'attirer en premier ceux pour qui la fiabilité est un élément critique, en vue d'acquérir la crédibilité suffisante pour que les autres viennent d'eux-mêmes[10]. Ainsi, les SSLL ont misé sur l'avantage technique de Linux afin de le commercialiser à des entreprises ayant un réel besoin d'une fiabilité extrême, comme celles de l'embarqué et des applications en temps réel (Maume [2002]). Le premier marché a donc été celui-ci. Il s'agit de réduire la source de RCA concernant l'aversion au risque des utilisateurs, en cernant d'abord les moins adverses d'entre eux. Ainsi, le fait que les grands comptes soient les premiers à faire le pas de l'adoption peut s'expliquer d'une part par le fait que ces entreprises disposent plus facilement des compétences internes permettant de faciliter la migration des serveurs sous Linux[11] et d'autre part par le phénomène de backward induction où les plus petites structures, sans en ignorer les avantages techniques, adoptent une position de suiveur.

[9]Ce que Farrell et Saloner [1988] formalisent avec la notion de «backward induction»

[10]Pour un modèle de diffusion technologique de Linux, on peut se référer aux travaux de Dalle et Jullien [2000] et de Jullien [2001]

[11]Ceci n'est pas sans rappeler les prémisses de l'informatique où seuls les grands comptes et les administrations étaient informatisés

Cependant, les acheteurs ne sont pas seulement sensibles à la fiabilité technique d'une solution, mais à son prix, élément déterminant dans le ralliement à Linux. Nombre d'entreprises ont adopté Linux en raison de son faible coût de licence. Bien que la majorité des SSLL adhère aux Libre en tant que principe éthique et voit dans la disponibilité du code source sa principale qualité, elles jouent sur la concurrence par les coûts pour favoriser la diffusion des logiciels libres. L'avantage concurrentiel en terme de coût dont bénéficient les SSLL, par la réutilisation (intégration, adaptation, ...) de logiciels dont la licence est libre, se répercute sur les services facturés aux clients, permettant de réduire les coûts de changements directs pour les adopteurs potentiels. L'économie en terme de coût apparaît ainsi comme un avantage facilitant la migration sous Linux et l'adoption des logiciels libres en général.

Les responsables des SSLL, interrogés sur la réaction des clients vis à vis des logiciels libres en général et de la licence GPL en particulier, confirment que les grands comptes sont les moins frileux à l'adoption des logiciels libres dans la mesure où leur préoccupation se situe uniquement au niveau technique. Si on leur démontre les avantages en terme de coûts de licence et de fiabilité, ces clients restent plus ou moins indifférents au type de licence alors retenue. Aujourd'hui, Linux semble pousser la porte des PME/PMI, mais le marché n'est encore qu'en phase d'émergence. De même que l'on cible les grands comptes et les administrations, l'ordinateur personnel n'intéresse pas (verrouillage plus important). Effectivement, bien que les discussions soient en cours sur l'intérêt de cibler le poste client, l'erreur qu'a fait Mandrake a justement été de miser principalement sur la convivialité et la facilité d'installation de Linux sur poste client, alors qu'il n'y avait pas de marché, d'autant plus que la simple vente de package n'y est pas véritablement profitable : les entreprises ont besoin de tout un panel de services informatiques (formation, maintenance, intégration, développement), tandis que l'utilisateur lambda ne demande que des produits finis, utilisables en l'état. Si la vente de licences d'utilisation apparaît très rentable sur les marchés de masse, de même qu'ils ne sont pas les clients des SSII, les particuliers ne sauraient être la cible des SSLL. Ainsi le marché visé est d'abord celui des grands comptes, avec une problématique orientée serveurs.

Si l'avantage concurrentiel en terme de fiabilité et de coût permet de cerner les segments de marché visés, reste à déterminer quelle est l'offre de services que les SSLL y proposent.

La politique-produit des SSLL : quels services sources de profit ?

L'alternative ne saurait passer par la rupture technologique puisqu'il s'agit de faire migrer des clients vers de nouvelles solutions équivalentes en termes de fonctionnalités, en jouant sur l'avantage concurrentiel de la fiabilité et du prix. Quelles peuvent être les sources de profit pérennes pour des sociétés de services spécialistes des logiciels libres, solutions marginales dominées par celles du modèle traditionnel des Éditeurs-SSII ?

La Syntec (Chambre Syndicale des SSII et des Editeurs de Logiciels), en vue de définir et d'identifier les SSII françaises, a dressé une nomenclature des services offerts par ces dernières. Elle recense trois catégories de prestations : 1) les prestations intellectuelles (recouvrant le conseil, l'audit, les études d'architecture de systèmes informatiques, l'intégration des systèmes, l'ingénierie, la maintenance, l'assistance technique et la formation) ; 2) les prestations machines (recouvrant les réseaux, les services liés à valeur ajoutée et l'infogérance) ; 3) les prestations de conception de progiciels (Mouline [1996]). À la vue de cette identification et compte tenu des activités identifiées lors des entretiens menés auprès de leur responsable, les SSLL peuvent être classées dans la catégorie des SSII car les services offerts sont très largement comparables. Cependant, les acteurs du libre semblent vouloir se démarquer des SSII. Pour ce faire, Alexandre Zapolsky, PDG de Linagora, a choisi de déformer le terme Société de Services et d'Ingénierie Informatique (SSII) en Société de Services en Logiciels Libres (SSLL)[12], afin de jouer d'une part sur la similitude (reconnaître

[12]Ce terme SSLL est une marque déposée par la société Linagora. Depuis, il est devenu le terme utilisé dans la presse informatique (*Le Monde Informatique* du 25 janvier 2002) et il commence à être adopté par l'ensemble des sociétés de service interrogées, c'est

que l'on est une société de services) et d'autre part sur la différence en affichant un domaine particulier d'expertise, celui des logiciels libres. Bien que le terme ne soit pas fédérateur, les responsables de ces sociétés insistent sur la nécessité de se positionner sur les logiciels libres et de l'afficher. En s'affichant sous le logo «SSII», les SSLL ne se démarqueraient pas des SSII et rencontreraient de plus amples difficultés dans la concurrence qui se joue déjà entre les deux types de sociétés. Les SSLL se positionnent sur les logiciels libres et espèrent ainsi attirer les entreprises intéressées par une migration sous Linux. En affichant leur rôle d'expert elles pensent qu'elles seront sélectionnées face aux SSII lors des réponses aux appels d'offre.

Effectivement, l'étude par entretiens des SSLL révèle que les principales concurrentes de ces sociétés ne sont pas les SSLL elles-mêmes mais bien les SSII. Si le marché du logiciel libre est émergent, au sens où peu d'acteurs sont présents, et si les SSLL sont en concurrence entre elles, cette concurrence semble pour le moment bénéfique car elle envoie un signal positif au marché, toujours selon l'idée de créer une aura professionnelle. Le fait qu'il y ait plusieurs acteurs présents sur le marché tend à rassurer les clients potentiels, les SSLL profitent donc de ces externalités positives générées par la crédibilité et par la simple présence de leurs concurrentes. De même, la raison d'être des SSLL n'est pas uniquement, aux dires des personnes interrogées, de «faire de l'argent avec le Libre», mais d'impulser un nouveau modèle production au sein de l'industrie du logiciel. Par exemple, Richard Stepniewski, directeur d'Adelux, interrogé sur le degré de concurrence entre les différentes SSLL, reprend le terme de «coopétition» pour signifier que s'il existe des concurrentes, on coopère souvent afin de crédibiliser l'ensemble des acteurs du Libre.

À la différence des SSII, les SSLL ne proposent pas d'activité pure d'édition. Si les SSII consacrent une partie de leur activité à l'adaptation de progiciels commerciaux existants (dans le cadre d'accords de licence passés avec l'éditeur) pour le compte de leur client, elles réalisent aussi le développement complet de solutions plus ou moins spécifiques. Parce que les SSII sont souvent centrées sur un métier particulier, elles développent, commercialisent et adaptent des solutions métiers (par exemple, un progiciel de gestion des collectivités territoriales). Elles se rémunèrent alors pour partie sur la vente de licences d'utilisation ainsi que sur la vente des services d'adaptation et d'évolution-maintenance. Dans le monde du libre, l'activité d'édition n'existe pas, les SSLL ne peuvent se rémunérer sur la vente de licences, car elles ne sauraient faire payer pour l'usage de ce qui est, par nature, libre d'utilisation. L'expérience l'a montré à la société Aurora qui a voulu, un temps, développer une offre «produit», vendant des licences en adoptant une activité d' «édition» de logiciels toutefois en GPL. Le problème auquel s'est heurtée la société est celui de l'incompatibilité des licences libres au modèle d'éditeur. Cumulant les pertes dues au manque de rentabilité de l'activité d'édition sur laquelle elle s'était centrée, la société s'est alors recentrée sur le service. Depuis, la société affiche de nouveau une rentabilité. Elle a d'ailleurs été acquise par Systran en août 2001. De même Jean-Pierre Laisné, fondateur de la société Linbox ayant déposée le bilan en juin 2001, et actuel directeur de la stratégie Open Source chez Bull, affirme, suite à son échec (l'idée à la base de Linbox était de vendre du produit sous forme de licences), que la seule stratégie de profit viable est de vendre du service. Dans le même sens, le récent recentrage des sociétés distributrices (RedHat, puis Mandrake) vers les activités de services (Lin [2002b] et Lin [2002c]) témoignent du fait que se cantonner à vendre des versions packagées de Linux ne semble guère profitable. Ainsi, le profit ne saurait naître de la commercialisation d'une offre standard ou packagée devant s'adresser à un marché de masse.

Pour concurrencer les grands acteurs en place, les SSLL affirment qu'il est nécessaire de miser sur l'extrême personnalisation des prestations offertes, en exploitant les économies d'échelle de développement dues à la libre réutilisation des composants logiciels. L'avantage de Linux et des modules libres est justement de permettre, par la mise à disposition du code source, cette très grande flexibilité. Les évolutions récentes de l'industrie du logiciel amènent Horn [2000b] à deux conclusions trouvant leur écho chez les professionnels rencontrés : l'inadéquation aux besoins des entreprises des progiciels commerciaux (où beaucoup de

---

pourquoi nous le retenons pour les qualifi er.

fonctionnalités sont inutiles donc inexploitées) et le besoin de services sur-mesure résultant de la diversité sans cesse croissante des utilisateurs. L'offre de services sur-mesure est permise par la modularité, modularité atteignant son apogée avec la logique du libre, puisque les composants logiciels eux-mêmes peuvent être modifiés. Si l'industrie du logiciel se dirige de plus en plus vers une logique de services (Eurostaf [2000]), les SSLL tirent partie de ce contexte. C'est uniquement sur les services misant sur la qualité et passant par une très grande diversité de l'offre en adéquation aux besoins très spécifiques des différents clients que la valeur peut se créer (Horn [1999]). Les SSLL misent sur la parfaite adaptabilité et interopérabilité des composants libres pour diversifier l'offre selon la demande. Les qualités du logiciel libre - fiabilité, flexibilité, portabilité et compatibilité - permettent de disposer de modules de grande qualité et d'optimiser les solutions pour les clients (Horn [2000b]). L'intérêt du développement libre consiste justement à disposer d'une base commune adaptable à chaque demande particulière. Lorsque l'on utilise des modules libres développés par d'autres, on gagne en temps, donc en coûts de conception, ce qui permet d'être plus rapidement rentable et de personnaliser l'offre à loisir. La valeur se crée en vendant non pas des licences, mais du temps et des compétences.

Il ne s'agit alors plus de vendre selon des objectifs de masse comme dans le cas de l'édition de logiciels, par l'exploitation des économies d'échelle, mais de profiter des économies de variété pour différencier son offre à partir d'une base commune de composants logiciels. La valeur ajoutée pour le client, source de rémunération pour les SSLL, se crée sur le caractère sur mesure des services et des développements logiciels. On ne se rémunère pas sur la vente de licences, mais sur celle du service. Il s'agit d'établir des contrats de confiance durables, basés sur la fidélisation des clients, par l'établissement de relations interpersonnelles.

La stratégie de profit identifiée lors des divers entretiens, commune aux SSLL, est celle de la diversité de l'offre fondée sur la qualité. Linux seul ne serait qu'un système d'exploitation de plus, mais la logique des logiciels libres va plus loin et les SSLL tentent de mettre en place un système de production alternatif au sein de l'industrie du logiciel.

Si l'on voit comment peut être appréhendée l'incertitude de marché, reste la question de celle du travail concernant la production même des biens et services offerts. Car rien ne garantit à l'entreprise qu'elle saura la mettre en œuvre la production des services ou qu'elle obtiendra de ses salariés l'activité désirée en temps voulu.

6.2 La mise en œuvre du systeme de production : Les spécificités organisationnelles des SSLL.

On a pu voir quels pouvaient être les segments de marché sur lesquels, compte tenu de leur environnement concurrentiel, les SSLL sont amenées à se positionner ; de même ont été identifiés les types de services qui semblent, en fonction des caractéristiques du bien «logiciel libre» et des priorités des acheteurs potentiels, pouvoir garantir la rentabilité de l'entreprise. Mais l'identification de la stratégie de profit d'une entreprise et de la politique-produit devant la satisfaire ne sauraient être suffisantes pour déterminer son système de production. Si ce dernier est perçu comme un «modèle d'entreprise» pour les responsables, ils doivent concrétiser leur projet en choisissant des moyens cohérents de mise en œuvre de la politique-produit. Ceci renvoie alors aux questions d'organisation productive, c'est à dire à celle des moyens et méthodes choisis pour conduire la production des services offerts (1) et de gestion de la relation salariale, s'exprimant par un compromis entre salariés et responsables afin d'obtenir des employés la production voulue (2). Ces deux éléments constituent des innovations de type organisationnel propres au système de production des SSLL. L'analyse se situe ici explicitement au niveau empirique.

6.2.1 L'organisation de la production de services : les SSLL, entre communauté et marché.

Beaucoup d'auteurs ont analysé le mode d'organisation des communautés (Lakhani et von Hippel [2000], Lerner et Tirole [2002], Jullien [2001][13]) et l'on serait enclin à imaginer ou à s'interroger sur la transposition du modèle d'organisation des communautés à l'intérieur des entreprises. S'il s'avère vrai que l'organisation de l'entreprise peut s'inspirer de celle des projets logiciels libres, l'idée de transposer le modèle d'organisation des communautés apparaît, à l'épreuve des faits, une utopie. D'après l'étude de terrain réalisée, l'élément clef de la réussite serait la considération et le respect des communautés, car le rôle d'expert des SSLL les conduit à se positionner en tant qu'interface entre communautés et clients. Si le modèle du don contre-don permet d'expliciter, en partie, le processus à l'œuvre dans les communautés (Lakhani et von Hippel [2000]), les SSLL ne sauraient s'y soustraire dans la mise en œuvre de leur production.

Le rôle d'expert des SSLL : une interface entre communautés et clients.

À l'heure actuelle, la sphère du Libre est peuplée de projets-logiciels développés par les communautés d'informaticiens en grande partie bénévoles et peu de logiciels libres sont le fruit de projets strictement commerciaux (Zope, GNAT, OpenCascade...) Il ne peut donc pas toujours y avoir un éditeur associé aux composants logiciels utilisés et réutilisés par les SSLL dans la mise en œuvre des projets de leurs clients. Ces sociétés doivent alors se positionner à l'interface entre les communautés de développeurs et les clients, quand les SSII se doivent d'être à l'interface entre éditeurs et clients. Le rôle des SSLL va être d'assurer l'expertise des logiciels libres pour le compte des clients, et, pour expliciter l'importance de l'expertise comme liant les communautés, les SSLL et les clients, nous pouvons reprendre les propos de Christophe Le Bars, fondateur et ancien directeur technique d'Alcôve. Pour lui, le rôle d'Alcôve, comme celui de toute SSLL, se définit comme celui «d'un médiateur entre la communauté et les entreprises», dans la mesure où il s'agit aussi de fournir des «services d'éditeurs» pour les logiciels libres. Dans ce sens, il rappelle que le métier des éditeurs ne se réduit pas à l'écriture de logiciels et à leur commercialisation, mais consiste aussi, et à hauteur de 50 % de leur activité, à fournir l'expertise sur leurs progiciels. Le but d'Alcôve est de se positionner sur ces «50 %» pour les logiciels libres. Effectivement, ces logiciels étant principalement l'œuvre des communautés, ils ne peuvent être rattachés à aucune maison d'édition qui en assurerait alors l'expertise, alors qu'existe, au sein de l'industrie du logiciel, un lien très fort entre les SSII qui jouent un rôle d'intégrateur - centrées sur le métier de leurs clients, elles répondent à leurs besoins personnalisés à partir de l'utilisation de briques logicielles propriété des éditeurs - et les éditeurs qui fournissent une partie de ces logiciels et en assurent généralement l'expertise technologique. Ceci ne signifie pas que les SSII ne proposent pas d'expertise technique pure sur des modules autres que les leurs, mais que ce n'est pas leur rôle premier alors qu'il doit être celui des SSLL puisque dans ce modèle, les éditeurs sont les communautés, non identifiables en tant qu'entité capable de fournir l'expertise en question. Les SSLL assurent un rôle de lien entre les communautés développant le code source des logiciels (code brut) et les utilisateurs utilisant ce code sous forme exécutable. Puisque les logiciels libres sont à la disposition de quiconque souhaite les utiliser, les entreprises pourraient puiser directement au sein des communautés (via les sites web des projets en question où même ceux des SSLL) ceux dont elles ont besoin, sans passer par le paiement d'une prestation de services. D'une part, les entreprises ont rarement, en interne, non pas les compétences, mais le temps pour analyser et déployer de telles solutions puisque ce n'est pas là leur cœur de métier. D'autre part, les modules libres sont disponibles sous forme brute, les contrats de licence, apparaissant comme convention structurant l'organisation du développement à l'intérieur des communautés, ne mentionnent en général aucune clause de garantie, ne serait-ce de bon fonctionnement. Il semble alors trop risqué pour une entreprise de bâtir son système informatique sur ces solutions *a priori* non éprouvées. Les SSLL assurent donc l'expertise pour ces logiciels afin de les rendre commercialisables, ce qui passe par l'ajout de clauses de garantie, la formation, la maintenance, etc., c'est-à-dire la «crédibilisation» de ces logiciels.

[13] De nombreuses références sont disponibles sur le site http://opensource.mit.edu/onlinepapers/.

Au-delà de la fourniture de l'expertise, les SSLL assurent les mêmes prestations que les SSII, à savoir l'intégration des solutions logicielles au sein du système d'information du client et le développement à façon. Avec la logique du développement logiciel libre, les modules étant ouverts et librement modifiables, les SSLL organisent la production à partir des composants mis à disposition par les communautés (Jullien [2001]). C'est en jouant sur la grande flexibilité permise par les licences libres, liée à l'interopérabilité des modules, que les SSLL peuvent résoudre le «paradoxe» de la modularité en oeuvre dans l'industrie du logiciel. À ce sujet, Horn [2000b], comme l'ensemble des responsables interrogés, rappellent les espoirs mis sur les bibliothèques de composants dans les années quatre-vingt dix. L'utilisation de briques logicielles pré-existantes devaient permettre de réduire le temps donc les coûts de développement logiciel, puisqu'il n'était plus nécessaire de partir de zéro, de réécrire l'ensemble du code source du logiciel. Mais dans cette logique, les briques ou composants restent sous licences propriétaires, les API (interfaces de communication entre différentes briques logicielles) demeurant la propriété des éditeurs. Avec les logiciels libres, tout devient accessible et l'on peut agir sur un composant librement en fonction des besoins des clients. Avec cette grande flexibilité, il est possible de faire du sur-mesure, donc de mettre en œuvre la politique-produit précédemment identifiée.

Dans le cas des biens informations comme l'est le logiciel, la présence de coûts de conception très élevés constitue une barrière à l'entrée rendant difficile la contestation du monopole en place, les possibles économies d'échelle qui en découlent (puisque les coûts marginaux de réplications sont eux quasi-nuls) étant identifiées comme la première source de rendements croissants d'adoption favorisant le verrouillage. En réutilisant le travail des communautés, les SSLL ne sont plus soumises au problème des coûts de conception puisque le développement des logiciels s'appuie sur l'existant[14]. La première source de RCA se voit alors neutralisée et les logiciels libres, parce qu'ils sont issus d'un autre mode de développement, peuvent se poser en alternative technologique aux standards du marché. La faiblesse des coûts de licence cumulée à celle des coûts de conception constituent alors un avantage concurrentiel pour les SSLL lors des réponses aux appels d'offre[15]. Mais l'utilisation pour sa commercialisation d'un travail bénévole réalisé dans la sphère hors-marchande ne saurait se faire sans la fourniture d'une contre-partie.

La nécessaire collaboration avec les communautés de développeurs et ses limites, intrinsèques au mouvement «logiciel libre».

Si dans une logique d'«édition», les SSII se doivent de payer à l'éditeur des royalties et autres droits d'accès pour utiliser les briques logicielles, ici la contre-partie se fait par contribution des SSLL au développement libre. Le but recherché par les communautés n'est pas la diffusion du logiciel mais l'écriture de celui-ci (Cohendet et al. [2001], [2002]) selon une méthode de développement logiciel favorisant la qualité du code source plus que la profitabilité selon la logique propre à l'édition commerciale (Mateos Garcia [2001]). Si la diffusion marchande d'un logiciel libre favorise la reconnaissance des développeurs ayant participé au développement (Lakhani et von Hippel [2000]), ce n'est pas là non plus la motivation première des développeurs qui cherchent avant tout à favoriser l'ouverture du code source comme des idées selon une éthique proche de l'éthique scientifique (Himanen [2001]). La contre-partie peut se faire par la contribution aux projets des communautés, qu'il s'agisse d'une contribution au sens technique par la participation au développement logiciel, ou de la contribution au projet plus large, en tentant de faire du Libre le mode dominant de développement logiciel.

Ainsi, et par respect de la GPL, les améliorations apportées par les SSLL se retrouveront dans «le pot

[14]C'est d'ailleurs ainsi qu'ont procédé les «distributeurs» de Linux.
[15]La facturation des prestations se fait généralement au temps passé.

commun» du Libre. Au-delà du simple respect des termes de cette licence spécifique, les SSLL peuvent contribuer au développement technique par la participation à divers projets, en y accordant une part de l'activité salariée dans le cas, par exemple, où elles spécialisent leur offre autour d'un logiciel libre. Elles signalent qu'elles ont alors un intérêt direct à participer au développement du logiciel en question afin de le rendre plus performant, donc plus attractif aux yeux du client, et à faire tester, par la communauté attachée au projet, les améliorations ou corrections apportées. La contribution passe aussi par la simple mise à disposition du travail que les SSLL réalisent sur certains composants libres, mais sur le site web associé projet, non seulement sur celui de la société et, surtout, en ne nommant pas ces développements (composants logiciels, patchs, etc.) du nom de la société, en raison d'une volonté d'affichage de transparence vis à vis des communautés. Souhaitant faire de ces logiciels les standards du marché, elles évitent d'y attacher le nom de la société afin qu'ils soient reconnus comme fruit d'un travail collectif et que les communautés continuent à participer. Certaines sociétés, comme Alcôve ou Easter-Eggs, disposent d'un site Internet (souvent un .org, différencié du .com attaché au site strictement commercial) spécialement dédié au mouvement du logiciel libre où l'on trouve des informations (articles, définitions), mais aussi des «laboratoires de développement» virtuels, avec les projets de développements propres aux SSLL sous licences libres[16]. Ceci répond à deux objets : d'une part, l'affichage, auprès des communautés, des contributions de la société, et d'autre part l'affichage auprès des clients, à qui l'on montre que l'on est vraiment immiscé dans l'univers des logiciels libres. Par ailleurs, si la contribution aux projets libres permet le «versement» d'une contre-partie à la communauté, elle constitue aussi une part de l'activité de recherche-développement des entreprises, permettant de maintenir le haut niveau d'expertise nécessaire. Effectivement, on a vu que les logiciels libres n'étaient rattachés à aucune maison d'édition pouvant en assurer l'expertise ou la formation. Pour proposer et fournir ces services de haut-niveau sur les logiciels libres, il est nécessaire d'auto-acquérir les compétences requises. La participation au projet des communautés semble être, aux dires des responsables des SSLL, un moyen idéal de maintenir le niveau d'expertise.

Dans cette logique, les responsables des SSLL signalent qu'ils ne considèrent pas le logiciel comme un bien mais comme de l'information (d'où le refus de la brevetabilité du logiciel). Ce qui est passible de protection n'est pas le code source mais les données propres aux entreprises, qui peuvent être protégées, par exemple, par la cryptographie. Les SSLL partagent la même culture et les mêmes principes éthiques que les communautés, leurs responsables comme leurs employés en sont d'ailleurs souvent issus. Isabel Zwegers, directrice d'Asynux, précise qu'il existe deux façons de contribuer à la communauté. Si la première passe par la collaboration «technique» aux développements libres, la seconde consiste à travailler dans le respect de l'éthique du libre. «*En créant une société sur ce modèle et en tentant de la crédibiliser aux yeux du marché par sa rentabilité, alors on peut démontrer quelque chose d'important, que l'éthique des communautés est viable au sein de la sphère marchande*». Souvent, il y a oscillation entre les deux voies : on ne participe pas directement aux projets des communautés, mais on met en GPL les projets de l'entreprise (Savannah chez Alcôve, Midgrad chez Aurora, Camtrace chez Axis et Agix, ...) On peut alors citer, parmi d'autres propos similaires, Jean Noël de Galzain pour qui «*si en théorie le logiciel libre permet (et les deux sont tout aussi importants) d'améliorer la rentabilité (productivité et compétitivité) et la qualité dans le service, il faut une entreprise pour démonter cela :que ce modèle existe et qu'il est viable*». Les principaux enjeux du libre, et les principaux défis pour les SSLL, confirme Alexandre Zapolsky de Linagora, ne sont pas technologiques, mais économiques et sociaux.

Cependant, le partenariat tacite entre communautés et SSLL admet un certain nombre de limites capables de le remettre en cause.
Les SSLL cherchent à favoriser la diffusion de leurs développements sous licence libre, mais elles doivent

[16]Philippe Breider, responsable commercial de la SSLL Easter-Eggs précise qu'en règle générale ces sites Web n'ont pas pour objet de créer une communauté de développeurs extérieure à l'entreprise et associée au projet.

répondre aux souhaits des clients qui sont encore réticents et demandent fréquemment des licences propriétaires. L'objectif premier est de faire adopter Linux, Apache, ..., les «standards» du libre, en en présentant les avantages déjà identifiés. Ensuite, il faut parvenir à ce que le client accepte que les développements effectués pour son compte soient protégés par des licences libres de type GPL. Cependant, les licences libres étant abusivement comprises au sens de «libres de droit», le client pense que le meilleur moyen de protéger les données de l'entreprise est de fermer le code source du logiciel, ce qui l'amène à retenir des licences de type propriétaire. On retrouve ici l'idée de lobbying que les SSLL doivent mener, aidées des communautés et des associations, auprès du grand public et des institutions nationales ou supra-nationales. Parce que les SSLL sont au service du client, l'expertise proposée comme le reste des services, y compris le développement à façon, ne peuvent se limiter à la sphère des logiciels libres. Il y a du propriétaire dans les SSLL car c'est le client qui décide[17]. La relation que les SSLL développent avec leurs clients peut alors être mal perçue par les communautés, parfois qualifiées d' «intégristes du libre», qui y voient la volonté de la part des SSLL de s'approprier leur code logiciel, ce qui peut venir contrarier la collaboration.
En outre, Mateos Garcia [2001] met en lumière la différence qui existe entre l'entreprise dont l'objectif est la profitabilité et les communautés dont le but est la qualité technique, considérant l'écriture logicielle comme un art. Les fins divergent et les objectifs des deux entités ne sauraient être les mêmes. Les SSLL ont à répondre aux besoins des clients, tandis que les communautés, répondent à leurs propres besoins. On note alors un manque d'applications métiers disponibles dans le champ des logiciels libres, donc un retour sur la question des économies d'échelle comme source de verrouillage conduisant à l'hybridation des systèmes informatiques, composés d'éléments libres et propriétaires. À ce niveau, l'«économie du Libre» apparaît encore trop jeune et trop dépendante du travail des communautés, le ralliement d'acteurs marchands, quant à lui, encore trop centré sur Linux.

6.2.2 Gérer l'incertitude du travail : un arbitrage entre règles et libertés.

Si l'on a pu voir quelle pouvait être le métier des SSLL et dans quelle mesure leur collaboration avec les communautés apparaissait primordiale dans la production des services, reste à s'interroger sur la mise en œuvre de cette production à l'intérieur des SSLL, c'est à dire analyser le rapport avec les employés. Rien ne garantit aux SSLL qu'elles obtiendront de leur salariés la production voulue en temps voulu (Boyer et Freyssenet [2000]), et il convient alors de questionner la mise en ouvre de la coopération entre communautés et employés en envisageant les spécificités du salariat «logiciel libre» conduisant au nécessaire établissement d'un arbitrage entre règles et libertés, compromis que nous pourrons illustrer en considérant l'exemple de la SSLL Alcôve.

Les spécificités du salariat du Libre.

Il ne s'agit pas non plus là de transposer le modèle d'organisation propre aux communautés de développeurs, mais de s'en inspirer car il faut tenir compte, d'une part, du lien entre ces communautés et les SSLL par la nécessaire collaboration analysée précédemment et, d'autre part, du fait que les salariés des SSLL sont souvent, si ce n'est issus de ces communautés, du moins de la culture de partage du code source. Des divers entretiens menés, il ressort qu'il n'y aurait donc pas de révolution dans la façon de travailler, mais la principale différence entre le modèle traditionnel et celui du Libre réside dans la présence de la communauté, ce qui engendre une autre façon d'organiser le travail.

Effectivement, le modèle de gestion salariale ne saurait être exactement le même que celui en œuvre dans

[17] Alexandre Zapolsky, PDG de Linagora, rappelle d'ailleurs à ce propos que le principal avantage de l'alternative proposée par les logiciels libres est de permettre au client de choisir entre des solutions équivalentes, libres ou propriétaires, plutôt que d'être enfermé par les seules licences propriétaires.

les SSII de type traditionnel puisque dans le monde du libre, ou conformément à «l'éthique des hackers», le maître mot est «show me the code». Les relations sont basées sur l'ouverture des idées et du code source, il ne saurait y avoir alors de rétention d'informations entre les parties. Richard Stepniewski, directeur de la SSLL Adelux, explique quelles sont les raisons humaines qui l'ont poussé à faire d'Adelux, au départ un département de la société Adequat (SSII), une entreprise juridiquement indépendante. La gestion humaine est différente, «*le personnel Linux ne se gère pas de la même façon que celui des SSII traditionnelles*», puisque dans les SSII l'organisation du travail est partiellement basée sur la rétention d'information ; les salariés ne diffusent pas spontanément ce qu'ils connaissent car c'est ce qui fait leur valeur intrinsèque et qui leur donne alors un avantage comparatif par rapport aux autres salariés. Richard Stepniewski affirme qu'il est difficile d'avoir les deux types de techniciens (SSLL et SSII) au sein d'une même structure. Dans les SSLL, note Jean-Noël de Galzain, PDG de la SSLL Aurora, on retrouve le modèle communautaire basé sur les deux notions de «partager et collaborer». De même, sachant qu'il n'existe pas d'éditeurs dans le Libre, il n'y a pas de session de formation organisée pour les salariés chez les éditeurs, comme on en trouve au sein des SSII. On retrouve le principe de libre circulation de l'information dans la mesure où chacun doit partager les connaissances acquises avec l'ensemble de ses collègues concernés par ces connaissances, ce qui peut donner lieu à des sessions de formations internes, alors plus formelles.

La contribution aux projets libres des communautés va permettre d'assurer en partie de la formation des salariés, alors perçue comme de l'auto-formation. Parce que les employés souvent issus des communautés, ils gardent des liens avec elles, ce qui rend plus aisée la collaboration pour les SSLL. L'auto-formation va se faire en partie durant les périodes dites «d'inter-contrats» correspondant aux «creux d'activité» du salarié entre deux affectations sur des projets. C'est lors de ces périodes que va aussi s'organiser la veille technologique. Cet élément semble commun à l'ensemble des SSLL même si certaines d'entre elles disposent d'un pôle spécifique de R&D.

La particularité des salariés des SSLL en tant que personnes issues de l'éthique des «hackers», permet, aux dires des responsables, un recrutement à niveau de rémunération en deçà du prix moyen observé sur le marché, la principale motivation des employés (comme des fondateurs) étant de travailler pour le logiciel libre, non de trouver le meilleur emploi en terme salarial. Cette caractéristique du salariat présente cependant un certain nombre d'inconvénients dans la mesure où il est possible de croire, pour les employés, que travailler dans la sphère des logiciels libres signifie travailler comme au sein ses communautés.

Au sein des communautés, les bénévoles travaillent pour le plaisir sur les parties du code qu'ils maîtrisent ou qu'ils affectionnent le plus, sans aucune contrainte de temps. Si les projets peuvent prendre une ampleur comme celle de Linux, c'est parce qu'il y a concentration de milliers de développeurs, que l'enjeu et l'effervescence sont suffisamment stimulants pour que chacun fasse au mieux, de manière à acquérir la reconnaissance de ses pairs. Au niveau des SSLL, les ressources humaines sont moindres, et l'on se doit d'être au service du client. On retrouve ici la principale différence entre les communautés au service de l'état de l'art, et les entreprises, visant la rentabilité. Car si les SSLL veulent démontrer que le Libre est viable en tant que modèle économique, il faut qu'elles le démontrent par leur propre rentabilité. Afin d'honorer, par exemple, les contraintes de temps dans la réalisation des projets et de favoriser les activités rentables pour la société, il est nécessaire d'imposer un certain degré de hiérarchie.

Le maître-mot qui revient lors de chacun des entretiens est celui de la «responsabilisation» des salariés qui serait la voie choisie pour gérer l'incertitude du travail, à l'opposé de celle de la division de l'intelligence du travail, historiquement dominante (Boyer et Freyssenet [2000]). Et ceci ne saurait se faire sans une certaine dose de hiérarchie visant à structurer l'entreprise ainsi qu'à cadrer les salariés. La libre circulation d'information doit, dans certains cas, être limitée aux frontières de l'entreprise, ce qui peut être compromis par le fait

que les salariés des différentes SSLL se rencontrent en dehors du travail et partagent les informations qu'ils détiennent, suivant la tradition culturelle qui est la leur. Mais la responsabilisation passe aussi par une souplesse dans l'organisation du travail, qui peut s'envisager différemment selon les SSLL, en fonction de leur taille et de leur orientation éthique. Il s'agit établir un contrat de confiance avec les salariés et de déterminer un arbitrage acceptable par tous, responsables et salariés, entre règles et libertés. Si le fait de laisser les employés participer librement à leurs projets personnels permet de gérer les inter-contrats et de maintenir un haut niveau d'expertise concourrant à accroître les performances de l'entreprise, l'instauration des libertés ne se fait pas uniformément ni universellement. Il existe un jeu de contre-partie entre le temps dont on dispose et celui que l'on doit. La participation aux projets des communautés, si elle permet le versement d'une contre-partie pour les développements récupérés, ne doit se faire que si elle apporte quelque chose à la société, pas uniquement au salarié. Il faut alors que les employés arbitrent d'eux-mêmes entre le loisir et le travail.

L'établissementdu compromis règles/libertés à travers l'exemple de la SSLL Alcôve.

Afin de mettre en lumière l'établissement du compromis entre règles et libertés, il est possible d'envisager sa réalisation en considérant l'exemple Alcôve.

Cette SSLL essaie d'établir un compromis entre règles et libertés à partir d'expérimentations, selon un processus d'essai/erreurs mis en œuvre depuis que la création de la société en 1997. Au départ, par exemple, il n'y avait pas d'horaires, les gens travaillaient comme ils le souhaitaient, la nuit, chez eux. S'agissant d'un travail purement intellectuel où il peut se passer de longues périodes de réflexion ou de recherche sans production à proprement parler, il est difficile de juger les employés sur leur travail au moment où ils le réalisent, et les responsables souhaitaient que leurs salariés travaillent là où ils se sentaient le mieux, afin d'accroître la productivité de la société. Avec la croissance de la SSLL, donc du nombre d'employés, et compte tenu des impératifs «clients» (un service d'assistance doit être assuré durant les «heures ouvrables»), des horaires ont été instaurés, le but étant par ailleurs de «socialiser» le travail, c'est à dire d'imposer aux salariés de se rencontrer afin qu'ils ne «s'enferment» pas, comme cela peut être le cas pour certains bénévoles œuvrant au sein des communautés. Cependant, Christophe Le Bars d'Alcôve affirme avoir conservé l'idée de n'exercer aucun contrôle sur ses salariés, sur ce que les gens font au moment où ils le font, les employés gérant leur temps comme ils le souhaitent. Les employés sont jugés aux résultats, une fois le travail accompli. Ils disposent d'une période de temps pour réaliser leur mission au terme de laquelle ils sont évalués. On compte alors sur la circulation de l'information pour la communication et la résolution des problèmes. Que le salarié contribue à des projets libres personnels sur son temps de travail n'est ni contrôlé ni sanctionné, car souvent il participe de même aux projets de l'entreprise sur son temps de loisir. La frontière est floue entre ce qui relève du travail salarié et du loisir, des contraintes et des libertés. La question de la flexibilité du travail, en terme d'horaires comme de l'organisation personnelle du travail, semble d'ailleurs être un point de convergence de l'ensemble des SSLL.

Entre 10 et 20 % du temps des salariés d'Alcôve serait consacré à la participation à des projets libres[18]. Christophe le Bars précise qu'il s'agit de 10 % à 20 % du temps de la société, pas de celui de chacun des salariés. Ce temps n'est pas distribué uniformément, mais en fonction de la confiance que l'on peut avoir dans certains salariés, des périodes d'inter-contrats et des aspirations de chacun. Il s'agit aussi d'une forme de récompense, la compensation d'un niveau de rémunération au-dessous du «prix du marché» pour les salariés de la SSLL Easter-Eggs, le fait de faire ce que l'on aime dans le cadre de son activité salariée. Mais l'intéressement, les mécanismes incitatifs, ne seront pas les mêmes pour tous. Dans le cas du recrutement des commerciaux, les SSLL cherchent avant tout des gens ayant l'expérience du métier, non les connaissances techniques qui font la valeur des informaticiens. Pour ceux-ci, les compensations

[18] R. [2001].

peuvent être monétaires. Par contre, à titre d'exemple, l'un des employés d'Alcôve et président de l'APRIL (Association pour la Promotion de l'Informatique Libre) dispose d'un aménagement particulier lui permettant d'inclure officiellement ses activités associatives au sein de son temps de travail. Ceci permet à Alcôve de bénéficier d'un écho favorable auprès de l'association, mais aussi de profiter des compétences du salarié alors que celui-ci affirme qu'il pourrait prétendre à un salaire beaucoup plus élevé dans une SSII traditionnelle, ce qui ne l'intéresse pas, la rémunération ne constituant pas une incitation suffisante à ses yeux[19]. Concernant les informaticiens, qui représente la majeure partie du salariat des SSLL, tous ne sont cependant pas des experts des logiciels libres, et même si on les y amène note Jean-Noël de Galzain, on essaie de recruter des gens compétents, ouverts et capables de travailler sur des technologies propriétaires. On trouve beaucoup d'anciens experts Unix (Axis et Agix) dans les SSLL, ces informaticiens semblant d'ailleurs s'orienter d'eux-mêmes vers Linux et les logiciels libres (Asynux), pour leur qualité de système ouvert ,dans la tradition du monde Unix. Le compromis entre règles et libertés que l'on retrouve au sein des différentes SSLL se base sur l'idée, explicitée par Christophe Le Bars, et confirmée par l'ensemble des responsables interrogés, que *«on n'accorde pas des libertés pour créer des conditions de travail géniales, mais pour obtenir une meilleure productivité, et on y arrive mieux avec une certaine dose de liberté»*[20].

Si les SSLL ont conscience que l'on peut générer des revenus avec les logiciels libres, leur objectif n'est pas de profiter des communautés sans contre-partie. De même, si elles croient au modèle de l'informatique libre, si l'utopie se vit au quotidien, il s'agit de démontrer pragmatiquement la viabilité du modèle, en affichant rentabilité et sérieux.

Conclusion

.

Les grandes lignes du système de production mis en œuvre par les SSLL françaises viennent d'être exposées. Aussi viable puisse-t-il apparaître au terme de ses premières années d'existence, l'avenir des SSLL semble incertain[21] et envisager une extrapolation de leur système de production au sein de l'industrie mondiale du logiciel apparaît prématuré[22]. Parce qu'elles sont le fruit entrepreneurial de l'éthique des communautés, les SSLL constituent un groupe d'acteurs économiques très particuliers en ce qu'ils tentent d'impulser un système de production exclusivement mis en œuvre dans la sphère du Libre. Ce courant minoritaire émerge dans un contexte semblant favoriser le modèle opposé du logiciel propriétaire par l'extension du champ de la protection de la propriété intellectuelle (reconnaissance de la brevetabilité du logiciel au niveau européen par exemple). C'est pourquoi, par leur adhésion aux associations de promotion et de défense du logiciel libre, et par leur participation à diverses manifestations à caractère parfois militant (Rencontres Mondiales du Logiciel Libre, ...), les SSLL participent du mouvement de lobbying, auprès des entités gouvernementales, pour la reconnaissance et l'utilisation des logiciels libres par les organisations publiques. Associations comme SSLL tentent alors de réduire l'incertitude de marché en agissant directement sur leur environnement institutionnel afin d'influence les décisions des acteurs au niveau micro-économique.

Si le Libre semble progresser sur ce terrain[23], il apparaît peut-être encore trop dépendant du travail des

[19] À ce propos, l'exemple de la SSLL Easter-Eggs peut être éclairant quoique constituant un cas très particulier où tous les employés sont propriétaires de l'entreprise (via une association) et perçoivent le même salaire, qui n'était que de 10 000 Francs nets en août 2000 (1524 €), de 12 000 F (1829 €) en décembre de la même année. Voir http://www.easter-egss.org.

[20] Afi n de valider les propos des responsables des SSLL, un questionnaire est actuellement en cours de développement devant donner lieu à une enquête précise auprès des salariés des SSLL

[21] Comme en témoigne le redressement judiciaire dont fait actuellement objet Alcôve, bien que le dépôt de bilan soit, principalement, le fruit d'erreurs de gestion de la trésorerie, et non de problèmes de rentabilité (*Le Journal du Net*, http://journaldunet.com).

[22] À ce titre, on peut souligner qu'une seule entreprise anglaise de type SSLL était présente à la LinuxExpo de Birmingham cette année, d'ailleurs d'une importance bien moindre que l'édition française.

[23] La Commission Européenne préconiserait dans un rapport (http://europa.eu.int/ISPO/ida/export/files/en/1115.pdf) l'utilisation des

communautés, ce qui pourrait conduire, au-delà de la simple cohabitation d'éléments logiciels libres et propriétaires, à l'hybridation des produits comme à celle des systèmes de production. D'une part, et comme cela a été précédemment souligné, il y a un fort manque d'applications métiers libres[24], ce qui engendre une hybridation des systèmes d'information des entreprises, composés de logiciels libres et propriétaires. Les responsables des SSLL précisent alors que, même si l'idée reste de faire du Libre un système de production complet, il y a des logiciels propriétaires dans leurs offres commerciales. D'autre part, le mouvement dit «Open Source» peut symboliser l'hybridation des systèmes de production eux-mêmes puisqu'il s'agit alors de l'assemblage d'un noyau «libre» avec des couches logicielles plus spécifiques et propriétaires. IBM, par exemple, met en œuvre ce type de stratégie en laissant le cœur du logiciel ouvert et en contribuant à son amélioration, tout en protégeant les couches logicielles plus spécifiques sur lesquelles la firme jouent de la différenciation pour se rémunérer.

Que le Libre soit encore en phase d'émergence, qu'il s'impose ou qu'il conduise plus vraisemblablement à une hybridation des modèles au sein de l'industrie du logiciel, l'observation des SSLL et de leur activité suggère la mise en place d'un nouveau système de production pas forcément contradictoire, mais complémentaire d'avec le précédent dans la mesure où il permet l'exercice du choix par les utilisateurs et le maintien de la diversité dans une industrie menacée de monopoles.

logiciels libres.

[24]Pour pallier ce manque, la Technopole de Soissons a lancé un concours de création logicielle dans différents domaines comme la gestion ou la comptabilité, voir http://soissons-technopole.org.

Les stratégies de libération du code source d’un logiciel par une entreprise : opportunités et difficultés. Premiers enseignements de deux exemples récents dans le secteur de la CAO et de la simulation numérique.

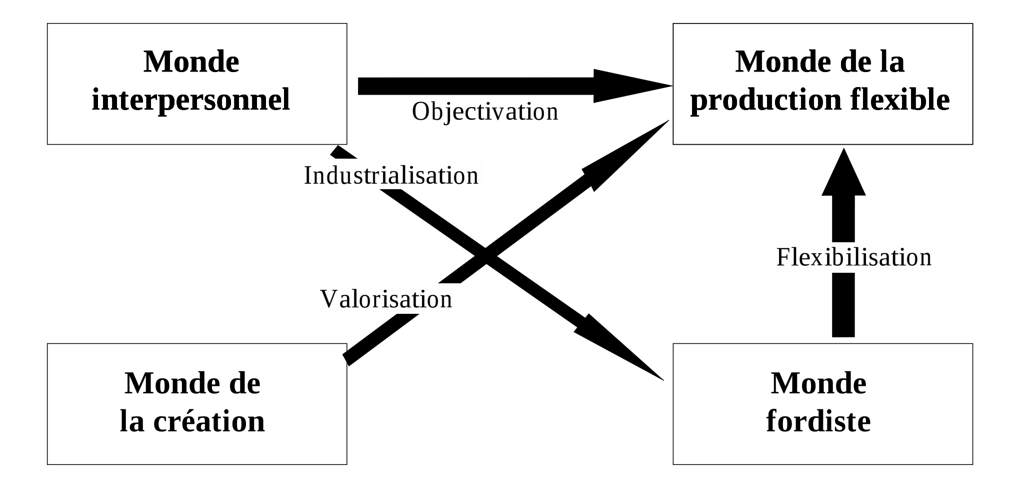
FRANÇOIS HORN,
Université Charles de Gaulle - Lille III, CLERSE - IFRESI CNRS, horn@univ-lille3.fr.

Indépendamment des progrès bien réels existants, la production des logiciels présente des spécificités qui rendent particulièrement difficile la réalisation de logiciels qui seraient conjointement très fiables, bien adaptés aux besoins des utilisateurs et produits avec de hauts niveaux de productivité. En particulier la résolution d'une de ces questions critiques de l'économie du logiciel rend souvent impossible des progrès identiques sur les autres dimensions. Il en résulte que dans l'économie du logiciel se développent simultanément des processus différents de rationalisation de l'activité, distincts selon les nécessaires compromis opérés entre les différentes caractéristiques de l'activité, ce qui confère à l'économie du logiciel une grande diversité.

Cette diversité a été analysée en termes de mondes de production des logiciels[1]. Chacun des mondes de production occupe une position particulière par rapport aux questions critiques de la production des logiciels (cf. tableau I). Le monde interpersonnel représente les difficultés à concilier productivité et fiabilité élevées dans la production de logiciels sur-mesure. Le monde fordiste des progiciels commerciaux se caractérise par une forte productivité mais une qualité souvent décevante, et ne peut répondre qu'à des besoins standard. Le monde de la création des logiciels libres apporte une certaine originalité et une efficacité indéniable sur certains segments de l'économie des logiciels, mais rencontre plus de difficultés pour s'étendre à l'ensemble des utilisateurs et des produits. Le monde de la production flexible des logiciels est porteur de potentialités pour concilier productivité et adaptation aux besoins des utilisateurs, mais il a du mal à se développer, surtout dans sa composante la plus prometteuse (la production de logiciels sur mesure à partir de composants standard).

Si la coexistence de mondes de production différents est un phénomène durable dans l'économie du logiciel, notamment en raison des très fortes complémentarités existant entre ces mondes, il n'empêche que l'économie du logiciel connaît d'importantes évolutions dynamiques. Ces changements se traduisent par l'existence de plusieurs trajectoires, correspondant au déplacement d'une production effectuée dans un monde de production vers un autre monde de production (cf. schéma I). À côté d'une trajectoire d'industrialisation, qui aboutit à une production en grande série de biens standard (monde «fordiste spécifique» des progiciels commerciaux), existent également des trajectoires de valorisation, de flexibilisation et d'objectivation, qui peuvent permettre un développement du monde de production flexible des logiciels, à partir des trois autres mondes de production.

Figure 7.1 **—** Les évolutions dynamiques de l'économie du logiciel.



Cette analyse a permis de mettre en évidence la domination actuelle des États-Unis sur l'économie du logiciel, grâce à une industrialisation beaucoup plus rapide de leur production.

Mais l'existence d'autres trajectoires, et notamment de celle que nous avons qualifiée de valorisation, peut

[1]Cette analyse est développée dans Horn [2000b].

	Absence de standardisation			**Standardisation du produit et/ou des composants**	
	Monde interpersonnel			***Monde de la production flexible***	
Produits dédiés	Type de produit :	sur-mesure ; service personnalisé.		Type de produit :	logiciels standards et services sur mesure. Logiciels sur-mesure à partir de composants standardisés.
	Exemple caractéristique :	application spécifi que.		Exemple caractéristique :	système de gestion d'entreprise.
	Producteurs principaux :	services informatiques internes, S.S.I.I.		Producteurs principaux :	S.S.I.I., sociétés de conseil, liées à des producteurs de matériel ou de progiciel.
	Utilisateurs principaux :	grandes entreprises, administrations.		Utilisateurs principaux :	grandes et moyennes entreprises, administration.
	Productivité dans la production des logiciels :	faible. Contradictoire avec fi abilité.		Productivité dans la production des logiciels :	assez élevée.
	Fiabilité des logiciels produits :	contradictoire avec la productivité.		Fiabilité des logiciels produits :	assez élevée.
	Adéquation aux besoins des utilisateurs :	variable selon le niveau d'intercompréhension.		Adéquation aux besoins des utilisateurs :	dépend de la qualité de la relation de service.
	Monde de la création			***Monde fordiste***	
Produits génériques	Type de produit :	logiciels libres (code source).		Type de produit :	progiciels commerciaux (code objet et services limités).
	Exemple caractéristique :	Le système d'exploitation Linux.		Exemple caractéristique :	tableur.
	Producteurs principaux :	universitaires, centres de recherche, créateurs indépendants.		Producteurs principaux :	éditeurs de logiciels.
	Utilisateurs principaux :	au départ communauté informatique.		Utilisateurs principaux :	ménages, entreprises (y compris P.M.E.), administrations.
	Productivité dans la production des logiciels :	variable.		Productivité dans la production des logiciels :	élevée.
	Fiabilité des logiciels produits :	amélioration rapide possible (pour les logiciels qui connaissent un succès initial.		Fiabilité des logiciels produits :	souvent insuffi sante.
	Adéquation aux besoins des utilisateurs :	forte pour la communauté informatique. Plus problématique pour les simples usagés.		Adéquation aux besoins des utilisateurs :	uniquement pour des besoins standards.

Tableau 7.1 — Les mondes de production du logiciel.

contribuer à modifier cette situation. De ce point de vue, le développement du monde de production des logiciels libres, outre sa contribution au nécessaire maintien d'une certaine diversité des progiciels, peut être riche de potentialités pour favoriser l'avènement d'un véritable «sur mesure de masse» dans l'économie du logiciel, notamment grâce à la production de composants logiciels libres. Toutefois, il n'est pas évident que l'extension du champ des logiciels libres se poursuive aussi spontanément[2] que cela a été le cas jusqu'à présent. L'analyse des différents facteurs du développement des logiciels libres montre que les facteurs défavorables à leur essor risquent de dominer les facteurs favorables.

En effet, premièrement, la motivation pour les acteurs du «monde de la création» à développer des logiciels libres risque d'être beaucoup moins importante pour des activités moins exaltantes et moins prestigieuses que les créations déjà effectuées[3], et pour le développement de produits dont les producteurs ne sont pas les utilisateurs principaux initiaux (logiciels de bureautique et de gestion, bibliothèques publiques de composants logiciels pour les applications métiers). Deuxièmement, la croissance des logiciels libres élargit le marché des producteurs de matériel et de services complémentaires. Mais cet élargissement peut entraîner l'augmentation des comportements de «cavalier libre», à savoir profiter de l'existence de logiciels libres sans participer à leur développement, en raison d'une série de phénomènes liés : apparition d'acteurs à la culture très éloignée du logiciel libre (par exemple IBM) à la différence des sociétés de services initialement présentes, transformation possible des valeurs culturelles de ces sociétés en conséquence de leurs succès[4], durcissement de la concurrence sur ce marché et vente à des utilisateurs plus diversifiés, moins sensibilisés aux enjeux des logiciels libres et pour lesquels le prix de la solution proposée est un critère déterminant, compétitivité supérieure des entreprises qui ne supportent pas de coûts de développement des logiciels libres. Troisièmement, l'extension du champ des logiciels libres à de nouveaux produits risque d'atténuer voire d'annuler le soutien que pouvaient apporter certains éditeurs de progiciels (souvent en position de faiblesse)[5] si les logiciels libres, de produits complémentaires à leurs propres produits, deviennent des produits concurrents. Quatrièmement, la menace que peuvent constituer les logiciels libres pour les intérêts de certains producteurs de progiciels commerciaux puissants, suscite des réactions, notamment par le dépôt de brevets, soit pour empêcher le développement des logiciels libres (Microsoft), soit pour les dévoyer en créant des logiciels qui n'ont guère de libres que le nom (Sun).

Il n'est donc nullement évident que continue à apparaître sur un rythme soutenu de nouveaux logiciels qui soient d'emblée des logiciels libres. Par contre, on peut se demander *s'il est possible que de nouveaux logiciels libres apparaissent à partir de la transformation par des entreprises en logiciels libres de logiciels qui étaient auparavant «privés»*[6]. *Quelles peuvent être les motivations de telles décisions ? Correspondent-elles à l'émergence de nouveaux modèles économiques ? Comment ces modèles s'insèrent-ils dans les évolutions plus générales de l'économie des logiciels ?*

Pour tenter de répondre à ces questions nous avons étudié deux expériences récentes (Open Cascade de Matra Datavision et Code Aster de EDF) qui sont successivement présentées dans la première et la deuxième partie. Ces expériences sont-elles des expériences marginales vouées à l'échec ou préfigurent-elles de nou-

[2]Jusqu'à maintenant le développement des logiciels libres, effectué selon une optique non marchande, n'a pas été non plus le résultat d'un projet explicite des pouvoirs publics, ce qui ne signifi e pas que les institutions publiques n'aient pas joué un rôle indirect important, notamment dans l'utilisation d'infrastructures matérielles publiques, de temps de travail de personnes payées par l'État, et dans la diffusion d'un modèle culturel propice au logiciel libre (de façon quelque peu schématique, le modèle de la recherche scientifi que publique).

[3]Le prestige d'une activité est notamment lié à son caractère novateur : développer un éditeur de texte programmable dans les années 70 (comme «Emacs» réalisé au MIT) pouvait contribuer à la renommée de ses auteurs ; ce n'est certainement plus le cas actuellement.

[4]Par exemple, Red Hat Software, principal «distributeur» de Linux, a été introduite à la bourse de New York le 11 août 1999.

[5]Sans faire de faux procès, on peut penser que la volonté de contrecarrer la volonté hégémonique de Microsoft a joué un rôle important dans certaines prises de position qui pourraient se modifi er en cas d'affaiblissement de Microsoft.

[6]Les logiciels qui ne sont pas libres sont souvent appelés logiciels «propriétaires». Le qualifi catif «propriétaire», largement utilisé en informatique, ne nous semble pas adéquat (on ne voit pas très bien de quoi pourrait être propriétaire un logiciel !) et il semble plus judicieux de parler de logiciel privé pour désigner un logiciel contrôlé par une entreprise qui en a la propriété.

velles évolutions significatives de l’économie du logiciel ? Resituer ces expériences dans le cadre de l’économie du logiciel analysée à partir de notre typologie en termes de «mondes de production» permet de dégager de premiers enseignements sur l’existence de nouveaux modèles économiques, les opportunités qu’ils peuvent présenter pour des entreprises confrontées à des situations particulières, tout en soulignant les difficultés que rencontrent des entreprises qui veulent changer de modèles économiques (troisième partie).

7.1 L’exemple d’Open Cascade.

Open Cascade est un logiciel qui se situe dans le secteur de la Conception et de la Fabrication Assistées par Ordinateur (C.F.A.O.). Ce secteur représente une part assez faible de l’économie du logiciel, mais a une importance stratégique croissante pour les entreprises utilisatrices de ces logiciels. Open Cascade est le premier exemple significatif de logiciel libre dans ce secteur. De même, c’est la première fois qu’un grand groupe industriel (E.A.D.S.) a pris la décision de transformer un progiciel commercial en logiciel libre.

Avant d’examiner les motivations de cette décision et les investissements qu’elle a impliqués (1.2), puis d’analyser les premiers résultats de cette stratégie (1.3), il est nécessaire de revenir brièvement sur l’historique de cette expérience (1.1).

7.1.1 Historique.

Qu’est-ce que Open Cascade ?

Open Cascade est une bibliothèque de composants logiciels (écrits en C++) qui servent au développement d’applications scientifiques et techniques. En 1996, elle comprenait 20 000 classes C++ avec par exemple quarante-trois façons différentes de trouver le centre d’un cercle. Le cœur de cette bibliothèque est un modeleur géométrique 3D qui est accompagné de bibliothèques d’échanges de données, de visualisation ainsi que d’un outil de développement permettant la paramétrage et la gestion d’attributs sur la maquette géométrique.

Cascade est le fruit du travail de trente personnes (le département de R&D de Matra Datavision) et représente huit millions d’euros d’investissement. À l’origine ces outils logiciels constituaient le cœur d’une plate-forme de développement destinée à élaborer le progiciel de C.A.O.CAO Quantum qui devait succéder à Euclid dans l’offre de Matra Datavision.

De Euclid au projet Quantum.

Il y a une dizaine d’années, Euclid occupait une place de choix sur le marché français des progiciels de CAO Mais son éditeur, Matra Datavision, se heurtait régulièrement à un syndrome classique des gros progiciels de l’époque, dit du «plat de spaghettis». À force de modifications, tout y était emmêlé, et il était impossible de changer quoi que ce soit sans faire bouger l’ensemble de manière imprévisible. Une grande partie des développeurs se consacraient à la maintenance d’un outil dont plus personne ne maîtrisait tout l’historique.

Pour préparer une prochaine génération d’Euclid (nommée Quantum) qui échapperait à cette dérive, Matra Datavision décida de développer ce qu’on appelait à l’époque un «atelier de génie logiciel» (AGL), et qu’on qualifie plutôt de «middleware» aujourd’hui.

L’objectif était de pouvoir développer du logiciel de C.A.O destiné à des applications scientifiques et techniques avec une grande réactivité à l’égard de l’évolution du marché et des attentes des clients. Ces derniers avaient de plus en plus besoin de calculer des modèles 3D complexes : pour visualiser le comportement d’un satellite, par exemple, il faut faire intervenir des compétences en thermique, en électricité, en radiations, etc. Afin de désenclaver les différents domaines technologiques, Matra Datavision avait imaginé de développer un

«fond de panier» sur lequel viendraient s'adapter des modules intercommunicants. L'idée se révéla excellente et allait bientôt être reprise par les grands concurrents de la société.

Après d'importants travaux de développement, Cascade, développé sous Unix selon un modèle «propriétaire» et formé d'un grand nombre d'outils et de composants réutilisables, se mit à mener une double existence : d'une part, Cascade était le cœur du logiciel Euclid (et de sa future version Quantum) et constituait les briques pour le bâtir ; d'autre part, Matra Datavision l'utilisait également pour une série de projets d'applications spécifiques, où la société était maître d'œuvre, et commença à le commercialiser à partir de 1995, sous le nom de CAS.CADE. Ses acheteurs l'utilisaient pour développer avec une grande productivité des applications de CAO pour eux-mêmes ou pour leurs clients et Cascade avait une base installée d'environ 130 clients actifs.

L'échec de Quantum et le tournant de Matra Datavision.

Par contre le projet Quantum allait être un échec pour Matra Datavision. Cet échec s'explique tout d'abord par l'évolution du secteur qui comme chaque segment de l'économie des logiciels subissait un fort mouvement de concentration à l'échelle mondiale en raison de la vigueur des rendements croissants d'adoption, qui condamnait peu ou prou les éditeurs moyens (Matra Datavision représente environ 5 % du marché mondial de la conception assistée par ordinateur) d'autant plus que l'existence d'éditeurs graphiques simples sous Windows rendait plus difficile l'accès au marché des PME. De plus au milieu des années 90, Matra Datavision commît une erreur de marketing en annonçant trop tôt le lancement de la nouvelle version Quantum, retardé par la volonté d'y intégrer de plus en plus de fonctionnalités, suscitant l'attentisme des clients. Les exigences d'un tel produit et le nombre d'ingénieurs de développement nécessaires grandissaient constamment tandis que les marges potentielles et les marchés diminuaient, sans compter les efforts pour développer un réseau de distribution.

En 1998, Hugues Rougier, nouveau P.D.G. de Matra Datavision, fît le constat que «le marché était désormais dans les services», en prévoyant que le marché des logiciels de CAO allait suivre la même évolution que le marché des logiciels de P.A.O. dix ans auparavant avec une division par dix du prix des fonctionnalités des logiciels. En conséquence, Matra Datavision fin 1998, décida de renoncer au modèle économique d'éditeur pour se recentrer sur la fourniture de services autour d'un progiciel concurrent, Catia, produit par Dassault Systèmes. Cette mutation s'accompagna du départ de 150 salariés et de l'embauche de 150 autres salariés. Matra Datavision comprend actuellement plus de 700 ingénieurs et consultants en Europe et en Amérique du Nord dans les domaines de l'audit et du conseil, de la mise en œuvre, de la formation, de l'ingénierie et du développement spécifique. La société est certifiée ISO 9001 pour la conception et la réalisation de services en ingénierie. Elle est devenue le partenaire commercial clé d'IBM pour le marketing de Catia et le leader mondial dans les ventes de la nouvelle version 5 de Catia.

7.1.2 La prise de décision et les investissements nécessaires.

Justification de la décision.

Dans cette évolution, qui n'a rien d'original - d'autres sociétés s'étant transformées d'éditeurs de logiciels en prestataires de services -, se posait le problème de l'avenir de Cascade.

Une étude de marché conduite auprès des utilisateurs avait montré qu'ils étaient très satisfaits de Cascade mais inquiets sur sa pérennité vu l'évolution de Matra Datavision. Cascade représentait un investissement important et un actif de qualité mais le développement commercial était très insuffisant avec un chiffre d'affaires d'environ 10 MF (1,5M €) : le prix de la licence avait dû être baissé de 1 million de francs (150.000 €)à 250.000 francs puis à 100.000 francs (38.000 puis 15.000 €).

Matra Datavision était convaincu que les métiers d'éditeurs et d'intégrateurs étant très différents, ils ne

pouvaient coexister dans la même structure. Souhaitant néanmoins pérenniser Cascade, en prenant en compte également la dimension sociale du problème, Matra Datavision prît la décision de le transformer en logiciel libre, sous le nom d'Open Cascade, dans un contexte où se développaient les logiciels libres et les sociétés constituées autour de ces logiciels comme MandrakeSoft, ce que confirmait une étude stratégique qu'avait fait réaliser Matra Datavision. Après s'être interrogé sur la pertinence de n'offrir en logiciel libre qu'une partie du logiciel, Matra Datavision décida de libérer l'ensemble du logiciel (sauf quelques composants avancés assurant la connexion avec les autres applications et la base de test) par crainte d'une réaction négative de la communauté du logiciel libre dont Matra Datavision escomptait le soutien. De même Matra Datavision choisît une licence qui impliquait de ne pas demander de redevances sur les développements réalisés à partir de Open Cascade avec comme objectif d'arriver à un standard de fait. En effet, l'objectif pour Matra Datavision était de compenser l'absence de commercialisation du logiciel par des ventes de services associés à Open Cascade (formation, assistance, conseil, développements spécifiques). Le processus de décision prît quelques mois jusqu'à juillet 1999.

L'amélioration du produit.

Avant d'annoncer la mise à disposition gratuite d'Open Cascade, en décembre 1999, il fallut encore engager de nouveaux coûts afin de porter le progiciel sous Linux, de renforcer sa modularité et d'améliorer son ergonomie. En effet, Open Cascade allait devoir «se débrouiller seul». Les utilisateurs, qui ne l'auraient pas payé, ne s'en serviraient vraiment que s'ils le maîtrisaient suffisamment vite. C'est ainsi que des travaux portant sur l'architecture d'Open Cascade furent effectués, la documentation complétée, et des exemples de programmation écrits pour toutes les fonctions importantes du logiciel. Et surtout, un module d'une grande puissance et qui représentait un travail important fut ajouté à la plate-forme : le module Application Framework (un serveur de déploiement gérant l'architecture applicative) qui fournit toutes les facilités pour bâtir rapidement une application complète, y compris interface homme-machine, visualisation et sauvegarde des données.

La filialisation d'Open Cascade.

Le lancement d'Open Cascade fut géré comme n'importe quel projet, y compris jusqu'à la protection contre d'éventuelles fuites : Matra Datavision voulait bénéficier du fait d'être la première entreprise à offrir en logiciel libre un outil de modélisation 3D et craignait qu'un concurrent n'effectue cette démarche avant eux. En mars 2000 un premier business plan fut lancé. C'est qu'en effet, outre les investissements initiaux pour améliorer le logiciel, d'autres investissements étaient nécessaires : soirée de lancement de Open Cascade, travail commercial d'explication envers les clients qui avaient acheté CAS.CADE, création de deux sites Web (un .com destiné à faire connaître l'offre et permettant notamment de télécharger le logiciel, et un .org destiné à animer la communauté des utilisateurs d'Open Cascade), développement d'un outil de CRM (Customer Relationship Management) spécifique permettant de gérer les clients potentiels que constituaient les personnes qui téléchargent le logiciel, développement de nouvelles versions de Open Cascade (après le développement d'une version pour Linux, puis pour Windows, sortie de la version certifiée 3.0 en Mai 2000, de la version 3.1 en novembre 2000 puis de la version 4.0). Au total ces investissements peuvent être estimés à environ 800.000 € en marketing, 200 à 300.000 € pour développer la version 4.0, 200.000 € en services divers, et 100.000 € dans le développement de l'outil de CRM.

Au départ Open Cascade était un département de Matra Datavision. Celle-ci, devenue une société de services ne voulait pas continuer à investir dans cette activité et souhaitait trouver des financements extérieurs par une entrée de Open Cascade sur le second marché. Mais la chute des marchés financiers en septembre 2000 et les exigences jugées exorbitantes d'éventuels partenaires (banques, sociétés de capital risque) firent que ce projet fut progressivement abandonné. Par ailleurs, la sortie d'un département de Matra Datavision

nécessitait un avis du Comité d'entreprise dans un contexte rendu conflictuel par les négociations sur un accord de réduction du temps de travail. Finalement ce n'est qu'au bout d'une année que Open Cascade a été filialisée sous son propre nom mais avec un capital détenu à 100 % par Matra Datavision.

7.1.3 Premiers résultats.

Un succès en termes de diffusion.

Incontestablement, la transformation de Cascade en logiciel libre est un succès pour sa diffusion et a redonné confiance aux utilisateurs dans la pérennité du produit. Open Cascade s'impose alors qu'il n'existe plus que trois plates-formes de ce type dans le monde. «On ne s'attendait pas à ce que le succès soit aussi gros» confie Hugues Rougier, PDG de Matra Datavision, d'autant plus qu'après des hésitations sur un lancement simultané aux États-Unis et en Europe, il avait été décidé de se concentrer uniquement sur l'Europe vu la très grande complexité du processus.

Au bout de la première année, douze mille utilisateurs de soixante-quatre pays avaient téléchargé le code source du progiciel et les sites Web avaient plus de 30 000 visiteurs par mois. Selon le responsable marketing, le nombre des utilisateurs de Cascade est passé de 100 à 2000 depuis qu'il a été libéré : 70 % des utilisateurs sont des éditeurs, qui n'ont souvent besoin que d'une toute petite partie du logiciel pour développer des applications spécifiques de CAO. Des sociétés importantes ont choisi Open Cascade pour développer leurs propres applications : par exemple, le CEA a élaboré ses propres outils de simulation nucléaire en environnement 3D à partir d'Open Cascade, Honda l'utilise pour concevoir ses motos, et Daimler-Chrysler pour contrôler la validité des pièces de ses véhicules. Pour les anciens utilisateurs de Cas.Cade, l'ouverture du code source présente des avantages : ainsi Optis, éditeur de CAO spécialisé dans les logiciels de simulation optique qui avait choisi Cascade (avant qu'il soit libéré) pour développer son logiciel Speos, estime que «depuis l'ouverture de Cas.Cade, le nombre de ses utilisateurs s'est multiplié, et, parallèlement, le nombre de questions soulevées sur le forum Open Cascade. Ainsi, nous sommes rapidement prévenus quand un problème est découvert, et ce dernier est plus vite réglé» (Le Monde Informatique, 25/01/2002).

L'utilisation avancée d'Open Cascade fait l'objet de cours au Pôle universitaire Léonard de Vinci en France et à l'University of California de Berkeley aux États-Unis qui a choisi Open Cascade pour le développement de ses applications CAO et décidé de contribuer à l'enrichissement du logiciel (publication de composants destinés à optimiser la triangulation des surfaces, nouvelle interface utilisateur). D'autres partenariats ont été conclus avec les universités de Karlsruhe, de Parme et de Tel Aviv, et le nombre d'utilisateurs universitaires est estimé à 250. Cette reconnaissance par la communauté universitaire est importante car elle permet d'enrichir Open Cascade des recherches effectuées dans ces institutions et elle garantit aux utilisateurs de pouvoir recruter des ingénieurs déjà formés à Open Cascade.

La gestion des activités de services.

Parallèlement la vente des services autour de Open Cascade a connu un certain essor et le chiffre d'affaires 2000 réalisé uniquement avec les ventes de services a dépassé de 55 % celui de 1999, qui incluait des ventes de licences. Le réseau de vente peut s'appuyer sur les cinquante ingénieurs commerciaux de Matra Datavision qui ont 4.000 clients avec Catia. Open Cascade a également renforcé son équipe interne, qui est passée de deux commerciaux en 2000 à sept commerciaux formés à Open Cascade actuellement, et développé des outils de monitoring des comportements des personnes qui consultent les sites et de gestion de la base de données de 15.000 contacts.

Open Cascade a tout d'abord commercialisé ses services sous la forme de trois packages (Silver, Gold, Platinum) comprenant différentes combinaisons prédéterminées de services divers (helpdesk, e-learning et formation, assistance technique, consulting, régie de développement, suivi personnalisé par un chef de projet).

Par la suite Open Cascade a innové en proposant trois nouvelles offres de support à la carte (Advantage, Premium, Excellence) : chaque offre est composée d'une certaine quantité de jetons consommables dans le trimestre et l'utilisateur peut dépenser librement ses jetons pour les services précis dont il a besoin, chaque prestation valant un certain nombre de jetons.

Le modèle économique.

Le modèle économique repose sur une nouvelle approche où c'est «c'est le savoir-faire autour du code plus que le code lui-même qui se monnaie» (Abou-Haidar, responsable marketing). Il est basé sur la distinction entre trois activités : le développement du logiciel lui-même, les services type éditeurs autour de ce logiciel (formation, intégration, maintenance) et les développements spécifiques réalisés à partir de ce logiciel. Comme le logiciel est fourni gratuitement, il faut en fait que ces deux dernières activités financent la première. Le raisonnement est le suivant : la libération du code source du logiciel doit permettre le développement continu d'un produit de qualité à moindre coût (a) qui peut être rentabilisé par un positionnement favorable sur les services associés (b).

a) Développement continu d'un produit de qualité à moindre coût

Le fait d'être un logiciel libre doit permettre d'améliorer le logiciel et de développer de nouvelles fonctionnalités : en raison de la disponibilité gratuite de la plate-forme pour créer des applications spécifiques et des possibilités de modification et d'extension de celle-ci en y ajoutant de nouveaux composants et de nouvelles fonctions, un grand nombre de spécialistes des logiciels vont élargir l'équipe restreinte de développeurs internes, Open Cascade fournissant des versions certifiées à intervalles spécifiques à une communauté internationale de développeurs. Dans cette optique, un comité de pilotage a été créé : il s'agit d'une équipe internationale associant les principaux utilisateurs industriels et universitaires dont le rôle est de mettre en œuvre des projets de développement par un réseau composé de développeurs de Matra Datavision et de développeurs extérieurs à la société.

En 2000, cinquante contributeurs extérieurs à Open Cascade ont fourni des apports divers : portage du logiciel sur d'autres systèmes (IRIX 64 bits, Alpha OSF), correction de défauts (fuite mémoire), traduction du tutorial en Espagnol... Actuellement il existe soixante-dix contributeurs actifs et l'objectif est d'arriver à cent. Ces apports extérieurs ne sont pas négligeables : Open Cascade estime qu'ils représentent environ 20 % de la valeur du logiciel. Selon Industries et Techniques (février 2001), Matra Datavision devait injecter environ 2 millions d'euros par an pour continuer à développer son outil et en 2000 l'entreprise a limité les frais à 1,2 millions d'euros. Toutefois, ces apports s'effectuent surtout à la périphérie du logiciel pas au centre : ils enrichissent mais ne consolident pas la plate-forme, et le développement de celle-ci continue à coûter cher à Open Cascade (environ 1 million d'euros annuellement). Ce coût est cependant minimisé par le recours à des informaticiens russes : la société aujourd'hui se compose de cent dix personnes, soixante-dix en Russie (à Nizhny Novgorod), trente-huit en France et deux au Japon. C'est qu'en effet, le prix de revient en France d'un chef de projets est de 650 € par jour et d'un développeurs de 450 € alors qu'en Russie il est à peine supérieur à 100 €.

b) qui peut être rentabilisé par un positionnement favorable sur les services associés.

(1) Les débouchés en termes de services.

Il existe deux grands débouchés pour Open Cascade, les services «type éditeurs» et les développements spécifiques.

La première catégorie englobe la formation, la maintenance et l'intégration. Cette dernière activité prend de plus en plus d'importance avec la continuité numérique dans l'entreprise du PLM (Product Life Cycle Management) qui fait de la conception du produit la base du système de gestion de l'entreprise. Open Cascade dispose d'atouts pour s'intégrer à d'autres logiciels dans la mesure où le code source est public et où, n'étant pas dominant sur son marché, les capacités d'interfaces ont été particulièrement développées. De ce point de vue, l'utilisation de ce logiciel garantit une certaine indépendance par rapport aux différents outils de CAO. À l'heure actuelle ces activités rapportent quelques millions d'€ qui ne couvrent pas les dépenses, mais le traçage efficace des utilisateurs n'est effectué que depuis fin 2000, ce qui devrait permettre de faire décoller ces activités.

Le deuxième débouché concerne des développements spécifiques effectués pour des éditeurs de logiciels (par exemple Samtech qui commercialise des solutions de calcul de structures) ou pour des entreprises industrielles (par exemple Mitutoyo un des trois grands fabricants de machines à mesurer qui souhaitait disposer d'une application permettant une confrontation entre le modèle numérique et le modèle réel avec intégration de la mesure de tolérances). Pour les développements spécifiques, l'activité se révèle très rentable avec des taux de marge supérieurs à ceux de Matra Datavision.

(2) Un outil de marketing pour des clients potentiels prêts à dépenser plus.

Pour la vente de ces services, le caractère libre du logiciel peut être considéré comme un outil de marketing. Les coûts de développement du logiciel peuvent alors être assimilés à des coûts de promotion («il faut se rattraper sur le complément, comme dans le cas d'un supermarché qui brade certains produits pour se rattraper sur d'autres produits» indique Hugues Rougier) ou à du sponsoring ("c'est une démarche du faible au fort" pour un logiciel qui n'est pas dominant et qui veut le devenir) : Open Cascade a fait l'objet de la publication de cent trente articles alors que, avec un budget important de communication Matra Datavision arrive à soixante articles par an.

La distribution gratuite de la version certifiée a fait croître sa diffusion et permet d'augmenter considérablement le cercle de prospects et de clients potentiels : «les utilisateurs viennent à nous» déclare Pierre Bruno, PDG de Open Cascade. La libération du logiciel est un moyen efficace d'accélérer commercialement l'accès au client avec l'atout supplémentaire que les contacts dans l'entreprise s'établissent avec les personnes directement intéressées par le produit. Or, les coûts commerciaux en avant vente peuvent être très élevés : pour Matra Datavision ils dépassent 10 millions d'euros par an. De plus comme les entreprises utilisatrices n'ont pas à payer de licence pour acquérir le logiciel, elles peuvent dépenser plus en services. Certes, comme l'indique Cirtes, société qui produit des corsets orthopédiques et qui utilise Open Cascade pour développer un logiciel de modélisation : «télécharger gratuitement un logiciel donne l'impression que tout est gratuit, mais il ne faut pas oublier que ce genre d'outil de développement vaut surtout par le support qui l'entoure» (Industries et Techniques, février 2001). Ceci étant selon Open Cascade, «la mise en œuvre de solutions open source, service compris, est 40 % moins chère que celle de produits propriétaire». La question pour Open Cascade reste d'arriver à transformer une partie des utilisateurs en clients des services proposés par la société.

(3) Avantage par rapport aux concurrents.

Dans le cas d'un logiciel libre, rien n'empêche une société qui ne participe pas au développement du logiciel libre (ou de façon marginale) de vendre des services autour de ce logiciel. À l'heure actuelle, mais l'expérience est très récente, Open Cascade subit une concurrence limitée sur les services qu'elle propose. Par contre, certaines sociétés, notamment indiennes, utilisent Open Cascade pour développer un logiciel de CAO bon marché qui pourrait concurrencer Catia pour les projets d'entrée de gamme.

En ce qui concerne les services proposés par Open Cascade, celle-ci ne semble pas trop craindre la concurrence d'autres sociétés qui pourraient être plus compétitives car elles n'ont pas les coûts de développement du logiciel à leur charge tout en pouvant l'utiliser. Hugues Rougier estime que les entreprises préfèreront recourir aux spécialistes de développement d'Open Cascade : il compare la situation à celle des cabinets de conseil

qui rendent publiques leurs méthodes, ce qui permet leur utilisation par des sociétés concurrentes, mais qui sont néanmoins choisis, car le fait d'être les auteurs d'une méthode apporte confiance et sécurité dans les prestations réalisées à partir de cette méthode. De plus Open Cascade estime qu'elle dispose de quelques années d'avance dans la gestion commerciale de la nouvelle activité qu'elle a «défrichée», à savoir proposer des services à des clients ayant téléchargé un logiciel libre. Elle a acquis des compétences dans la compréhension fine des besoins du client, dans la capacité à faire des propositions rapides et ciblées, dans le management contractuel du client, dans la gestion de projet à distance et dans la gestion de ressources offshore avec la possibilité de ventes aux clients en régie, et a élaboré des outils adéquats, notamment par le développement d'un logiciel de CRM.

Les difficultés rencontrées.

Toutefois dans la mise en œuvre de ce modèle économique Open Cascade a rencontré certaines difficultés. Ces difficultés se concrétisent dans les résultats actuels et dans les prévisions à court terme de la société : en 2000 le chiffre d'affaires était de 2.100k € avec une perte de 2.300k €, en 2001 le chiffre d'affaires était de 3.500k € avec une perte de 2.000k €. En 2002 la société a pour objectif un chiffre d'affaires de 4.200k € et une perte de 1.000k € et en 2003 l'objectif de chiffre d'affaires est de 6.000k € avec l'objectif d'atteindre l'équilibre.

Certaines de ces difficultés peuvent être considérées comme conjoncturelles : ainsi en 2000 la révision à la baisse de certains investissements dans les technologies de l'information et de la communication en raison de l'évolution de la situation économique a entraîné le report ou l'annulation de quelques projets importants pour Open Cascade[7]. L'impact de ces événements sur Open Cascade est significatif dans la mesure où on se situe dans un univers de contrats très techniques avec un cycle de décision particulièrement long.

Il faut également tenir compte de certaines erreurs initiales qu'estime avoir commis Open Cascade dans la gestion de cette nouvelle activité. Celle dont les conséquences ont été les plus dommageables est le fait qu'au départ, n'étaient pas identifiées les personnes qui téléchargeaient le logiciel, les réticences à demander l'adresse email s'expliquant par l'éthique originelle d'Internet.

La plus grande difficulté provient de la véritable mutation culturelle que représente le changement de modèle économique pour une structure dotée d'une certaine inertie due à son histoire. Il n'est pas évident de transformer un ancien département de R&D en une entreprise commerciale dynamique. C'est qu'en effet la réussite de l'entreprise présuppose d'arriver à transformer à moyen terme quelques dizaines de milliers d'utilisateurs en 200 clients. Or, la société est seulement passée de 17 clients en 2000 à 30 clients en 2001 avec un objectif actuel, non encore atteint, de 70 clients. Si la libération du code source a effectivement permis d'accroître considérablement - bien au delà des prévisions initiales - le nombre d'utilisateurs, il apparaît qu'il est par contre beaucoup plus difficile que prévu d'avoir une proportion même minime de ces utilisateurs qui achètent des services à l'entreprise. Les utilisateurs s'habituent facilement à la gratuité du logiciel mais admettent plus difficilement de devoir payer les services proposés. Pour les prestations les plus banalisées (help desk par exemple), il existe même une relative contradiction entre les deux objectifs d'extension de la diffusion du logiciel et de conquête de clients : en effet, la réussite de la diffusion du logiciel suppose que celui-ci puisse être utilisé facilement, ce qui a été réalisé par son amélioration significative et par le bon fonctionnement du forum de discussion, mais qui rend moins nécessaire l'achat de certaines prestations, sauf à assurer une très grande qualité à ces prestations (garantir à l'utilisateur une réponse rapide et précise à son problème) qui de ce fait sont coûteuses. À l'heure actuelle, ces difficultés conduisent l'entreprise à réfléchir à la vente des versions certifiées du logiciel, les versions en cours de développement restant librement téléchargeables.

[7]En 2001, selon Gartner-Dataquest, le budget pour l'équipement dans les TIC (ordinateurs, serveurs, stockage, logiciels, télécoms) des entreprises a été réduit de 8,4 % (premier recul depuis 1958) alors qu'il progressait de 10 % à 15 % par an dans les années 1990.

7.2 La libération du code source du Code Aster d'EDF.

Cette expérience, à bien des égards différente de la précédente ne serait-ce que dans les objectifs poursuivis, est encore plus récente. Avant d'en examiner les tout premiers résultats (2.2), il est nécessaire de revenir sur l'historique du Code Aster qui permet de comprendre pourquoi E.D.F. a pris la décision de transformer ce logiciel en logiciel libre (2.1).

7.2.1 Historique.

Après avoir expliqué ce qu'est Code Aster (1), nous verrons en quoi l'échec des tentatives pour commercialiser ce logiciel (2) a incité E.D.F. à le transformer en logiciel libre (3) .

Qu'est-ce que Code_Aster ?

Code_Aster est un logiciel de simulation numérique qui facilite la mise au point et la maintenance des installations de production et de transport d'électricité. Il a été conçu pour répondre aux besoins du groupe EDF en analyse mécanique et en simulation multi-physique. En effet, les exigences de sûreté et de disponibilité des équipements de EDF nécessitent d'étayer les décisions de mise en exploitation, de réparation ou de remplacement par la simulation numérique. Une de ces fonctions essentielles est d'évaluer le vieillissement d'installations de distribution et de production d'électricité, y compris nucléaires.

La diversité des ouvrages gérés par E.D.F. (260.000 supports de ligne, 300 groupes turbo alternateurs, 213 barrages, 58 unités de production nucléaire) et la diversité des matériaux utilisés font que ce logiciel permet d'aborder un vaste domaine de simulations, tel que la thermo-mécanique, la 3D linéaire et non linéaire, la statique et la dynamique, la chaudronnerie, les machines, le génie civil et la géométrie complexe des milieux poreux.

Son processus de développement et de diffusion s'appuie sur un système de qualité éprouvé et audité. En effet, les exigences de qualité du Code Aster sont particulièrement fortes (un code de qualité «ingénierie nucléaire») dans la mesure où il est utilisé pour justifier de la durée de vie des ouvrages aux Autorités de Sûreté. Ce logiciel a été développé par le département R&D d'EDF depuis 12 ans, une dizaine de développeurs y consacrant la majorité de leur temps. Il comprend 800.000 lignes de code gérées en configuration, 1170 tests de non régression et de qualification, et 9000 pages de documentation (5 manuels en 9 tomes). Il est exploité par plus de 190 utilisateurs en permanence, dont une quarantaine sont des industriels ou des chercheurs n'appartenant pas à EDF. À EDF, il a été utilisé pour effectuer 35000 heures de calcul en 2000, avec 16 calculs supérieurs à 2h chaque jour.

L'échec des solutions commerciales.

Code Aster est un des résultats de la politique ambitieuse de EDF de maîtrise complète de la simulation numérique dans les différents domaines résultant de son métier principal de producteur et de distributeur d'électricité : mécanique des fluides, neutronique (cœur de réacteur) et plus tardivement mécanique des solides. Cette politique volontariste de «grand code» s'est traduite en 1988 par la décision de développer un code unique de mécanique des structures, le Code Aster. Un tel effort d'investissement logiciel est devenu une particularité d'EDF : elle est un des derniers industriels européens à effectuer de tels investissements, les autres industriels achetant leurs outils auprès d'éditeurs américains.

Rapidement EDF estima qu'elle avait intérêt à «valoriser» un tel investissement mais rencontra des difficultés pour élaborer un modèle commercial de vente du logiciel : en effet, EDF ne pouvait devenir éditeur de logiciel parce que son statut encadré par le législateur ne l'autorise pas à développer certaines activités (par

exemple les services informatiques), et parce que l'extériorité de son métier principal la prive de la légitimité d'un éditeur de logiciel : une équipe de R&D diffère fortement de l'assistance client. Certaines expériences de création de société (par exemple, Simulog, émanation de l'INRIA, autour des codes de thermohydraulique et de mécanique des fluides) pour effectuer de la valorisation externe internationale avec du support s'avéraient peu concluantes.

EDF tenta néanmoins de vendre indirectement des licences de Code Aster (au prix de 16.7k €) en s'appuyant sur trois distributeurs et prestataires de services avec l'espoir de créer une synergie avec ces sociétés : CS (ex CISI) éditeur de logiciel scientifique visant les PME, CETIM qui apportait la caution technique des industries mécaniques à visée scientifique et SAMTEC société belge éditrice de logiciels et prestataire de services dans les domaines de l'aéronautique et de l'espace. Une faiblesse de Code Aster était le manque de modules de pré-traitement et de post-traitement permettant le transfert entre les données pour la simulation numérique et la CAO. Samtec avait dans ce domaine une stratégie en s'appuyant sur Cascade et il était prévu d'engager un programme de mutualisation des développements à base de Cascade pour enrichir le logiciel. Mais la décision en décembre 1998 de Matra Datavision de mettre fin à son activité d'éditeur de logiciel et de se transformer en société distributrice de Catia et prestataire de services autour de ce logiciel, entraîna une grande incertitude sur l'avenir de Cascade et l'abandon de ce projet. EDF essaya de susciter d'autres regroupements autour de solutions moins ambitieuses mais les stratégies difficilement conciliables des différents partenaires empêchèrent ces projets d'aboutir. En 1999, EDF se retrouva dans une situation de grande perplexité : les mécanismes de distribution ne fonctionnaient pas bien, il existait des problèmes d'adaptation aux clients qui se retournaient vers EDF dont ce n'était pas le métier et les ventes de licence ne rapportaient rien. C'est dans ce contexte que Matra Datavision prît la décision de transformer Cascade en logiciel libre, ce qui ne fut pas sans conséquence sur la décision de EDF.

Le processus de décision.

En septembre 1999, EDF décida de renoncer à investir dans le développement de modules de prétraitement et de post-traitement, en considérant que ce n'était pas son métier, et en espérant pouvoir s'appuyer sur des solutions inventées par d'autres. Il apparaissait clairement que cela équivalait à renoncer à une diffusion commerciale de Code Aster.

Fin novembre, un comité de pilotage prît la décision d'instruire le modèle de diffusion du logiciel libre. En juillet 2000, à partir d'un état des lieux du secteur commercial et d'un bilan avec les distributeurs, EDF décida de mettre fin aux contrats de distribution et de ne plus instruire le modèle commercial et d'approfondir l'étude du modèle du libre : évaluation des risques, aspects juridiques et choix de la licence, contraintes de réalisation des sites Internet, pertinence économique du modèle, opportunité d'être le premier à proposer en logiciel libre une application complète en simulation, possibilité de trouver des contributeurs extérieurs au développement (nécessité de convaincre quelques industriels et centres techniques d'être des contributeurs permettant de créer un comité de pilotage externe).

Il fut décidé de réaliser de toute façon des sites Web sur les logiciels Code Aster, avec une utilisation Intranet permettant d'améliorer la circulation des informations internes à EDF, une utilisation Extranet pour les fournisseurs d'EDF utilisateurs du logiciel et une utilisation Internet avec un site indépendant du site institutionnel EDF. L'investissement s'éleva seulement à $\frac{1}{2}$ homme / an mais un investissement supplémentaire serait nécessaire pour créer des versions anglaise et espagnole du site.

La recherche d'acteurs pouvant constituer un premier cercle de contributeurs externes s'appuyait sur quelques partenaires intéressés (Institut Français du Pétrole, Michelin et IFP, une société qui commercialise des logiciels et des services). Les changements permanents de stratégies de ces partenaires dans un contexte très incertain empêchèrent la réalisation d'un accord simultané avec les différents partenaires pressentis et conduisirent à l'abandon momentané du projet de premier cercle.

En mars 2001, le site Internet était opérationnel ; après des discussions avec des membres de la communauté du logiciel libre et avec Open Cascade, les projets de décision et les projets de licence étaient prêts avec l'avis des juristes. La licence choisie fut la licence GPL (GNU General Public License) certifiée par la Free Software Fundation qui permet aux utilisateurs de développer du code complémentaire, de construire des applications commerciales spécifiques et de proposer des services autour du logiciel. Les différents acteurs d'EDF étaient convaincus de la pertinence du projet et la décision fut prise le 6 juillet avec la volonté d'aller vite pour capitaliser l'atout d'être le premier. Le 19 octobre, EDF mettait à disposition Code-Aster en logiciel libre sur son site Internet.

7.2.2 Les premiers résultats.

Un succès en termes de diffusion.

100 jours après l'ouverture du site, celui-ci avait fait l'objet de 14 000 visites dont 65 % provenant de France, 39 % des États-Unis (connexions en général courtes vu que le site est exclusivement en français) et 6 % de 32 autres pays. Il y avait 440 téléchargements de l'exécutable (87 % de la version Linux, 13 % de la version Solaris Irix, et il fallut attendre 2 mois pour avoir une demande d'une version sous Windows NT). Seuls 50 % des téléchargeurs avaient été identifiés dans la mesure où l'identification est facultative. Parmi les téléchargeurs identifiés, il apparaissait que 60 % venaient du monde académique et 40 % de l'industrie, et que 83 % des téléchargements s'effectuaient en France, les autres se répartissant entre 20 autres pays. Le code source n'a été mis en ligne que trois mois après et a été également massivement chargé (140 téléchargements en 40 jours). Les initiateurs du projet avaient quelques craintes sur le démarrage des forums mais les échanges se révèlent être de bonne qualité. Si les réponses sont essentiellement l'œuvre des salariés d'EDF, elles sont rapides et assurées spontanément par les différents participants au projet sans avoir eu besoin de définir des astreintes.

Ce succès est en partie due à la publicité qu'a donnée la «communauté du libre» à la décision d'EDF. Selon Jean-Raymond Lévesque, l'initiateur du projet, «nous avons bénéficié d'une aide fantastique des acteurs du libre», ainsi que de l'existence de jeunes informaticiens favorables aux logiciels libres, notamment dans les entreprises industrielles.

On commence à voir apparaître des demandes pour la fourniture de services, mais dans ce domaine les objectifs d'EDF sont bien différents de ceux de Open Cascade.

Des objectifs différents.

L'objectif d'EDF est l'amélioration du logiciel pour mieux répondre à ses besoins internes. La diffusion externe de Code_Aster par Internet en «logiciel libre» doit permettre d'agrandir le réseau d'utilisateurs, et donc favoriser une mise à l'épreuve du logiciel auprès d'un vaste public de spécialistes très impliqués, qui pourront également conforter l'avis des experts nommés pour valider le code avant sa mise en production. En effet, «la valeur d'un logiciel de simulation et d'étude repose largement sur la compétence et l'esprit critique de ses utilisateurs» (entretien avec Jean-Raymond Lévesque). Le but est d'obtenir une «reconnaissance par l'usage» du logiciel, qui favorise la détection des anomalies, la comparaison avec les autres produits existants et la validation du logiciel pour des utilisations où l'évaluation des risques génère des exigences de transparence de plus en plus importantes : «la validation des nouvelles fonctions de Code_Aster, avec le niveau exigé dans le secteur nucléaire, est difficile avec le seul usage interne du logiciel. Nous disposons en effet d'un nombre d'utilisateurs insuffisant pour qualifier rapidement les nouveaux développements» (Jean-Raymond Lévesque, Le Monde Informatique, 1/2/2002). Cette amélioration incrémentale du logiciel facilitée par son caractère libre est une question décisive pour un logiciel très technique où les temps de diffusion sont extrêmement longs (un logiciel de simulation numérique développé 12 ans auparavant est considéré comme un produit «jeune»)

et où les tentatives de rénovation globale de l'architecture sont très difficiles (une tentative d'architecture révolutionnaire développée autour de l'Université de Berkeley s'est soldée par un échec).

Pour ce faire, il s'agit d'amorcer un cercle vertueux fort entre qualité croissante et engagement des utilisateurs dans une logique «gagnant /gagnant» permettant de bénéficier de la dynamique d'un réseau que l'on contribue à développer. La croissance du nombre d'utilisateurs peut s'effectuer dans plusieurs directions grâce au caractère libre du logiciel. Les fournisseurs d'EDF, confrontés à une politique d'achat qui se durcit avec une montée des exigences sans garantie de marché et une pression par les prix, bénéficient d'une licence gratuite d'utilisation (exploitation et développement) de Code Aster pour effectuer d'autres travaux pour des tiers. Cette opportunité peut également profiter à d'autres industriels et fournisseurs de services en simulation numérique, ce qui favorise des retours d'expérience externe issus d'utilisations hors du contexte EDF. Un deuxième axe de développement concerne les équipes de recherche et agences de développement, qui peuvent être intéressées par la libre disposition d'une plate-forme de simulation 3D avec plusieurs points d'entrée, indépendante des solutions commerciales, et permet notamment d'accélérer les épreuves de la version de développement. Le libre accès au logiciel, à sa documentation théorique et à un corpus de tests et d'exemples peut également faciliter l'utilisation de Code Aster dans la formation, ce qui entraîne l'existence d'un «vivier» d'acteurs formés, atout important pour la diffusion ultérieure du logiciel.

Si l'objectif principal d'EDF est l'amélioration qualitative du logiciel à partir des remontées d'expérience suscitées par une utilisation plus importante et plus diversifiée du logiciel, EDF espère également enrichir le logiciel des contributions d'une large communauté de développeurs : «l'objectif est de baisser les coûts informatiques tout en favorisant le développement coopératif, mutualisé entre plusieurs entreprises» (Jean-Luc Dormoy, Le Monde Informatique, 1/2/2002). EDF souhaite que sa démarche puisse capitaliser les contributions de nombreuses équipes de recherche (universitaires ou industrielles) et leur offre en retour une structure d'accueil pérenne réutilisable facilement pour leurs propres travaux : «nous espérons que le nombre de contributeurs doublera d'ici un an. Un développement collaboratif bien encadré permet d'atteindre un meilleur niveau de qualité qu'une équipe isolée» (Jean-Raymond Lévesque, Les Échos, 19/11/2001).

Par contre, et c'est une différence importante avec Open Cascade, EDF n'a nullement l'intention de commercialiser des services autour de Code Aster : «on veut voir émerger un pôle services mais sans nous» affirme Jean-Raymond Lévesque. Dans le communiqué rendant publique la décision de faire de Code Aster un logiciel libre, EDF précise que «tout organisme peut proposer avec des conditions définies par lui et sans la caution d'EDF des services pour l'exploitation du logiciel (formation, support au sens large, portage, qualification) et pour l'enrichissement du logiciel (formation, support, distribution particulière)». Convaincu qu'il existe une synergie forte entre diffusion du logiciel et création d'un réseau de sociétés de services autour de Code_Aster, EDF a apporté une aide au démarrage pour ces prestations en proposant des formations gratuites à l'utilisation et au développement avec les auteurs du logiciel et une assistance téléphonique pour les premiers organismes intéressés.

Les perspectives.

Après une première vague de curiosité, la fréquentation des sites continue à croître notamment dans le monde industriel. Des enseignements sont prévus l'année prochaine sur Code Aster. Pour les développeurs d'EDF, le premier bilan est positif («cela marche et c'est utile») grâce notamment à l'utilisation de Code Aster dans des domaines insoupçonnés. EDF continue le développement pour ses besoins internes et a comme perspective de terminer la version 6 du logiciel pour octobre 2002 et de réaliser la version 7 pour octobre 2004. EDF entend continuer à développer et à animer les réseaux pour faciliter les apports de contributeurs externes universitaires et industriels. Une Assemblée Générale des utilisateurs est prévue en octobre 2002 avec un atelier «utilisation» pour effectuer un bilan des usages et un bilan des services et des distributions, et un atelier «contributions» pour l'adoption d'une charte de bonne conduite et la constitution d'un comité de pilotage.

À terme, EDF n'exclut pas que le centre de gravité du produit puisse migrer vers l'extérieur, ce qui allégerait la charge financière pour EDF tout en préservant le produit et les emplois.

7.3 Premiers enseignements.

Il est évident que le nombre limité d'expériences et leur caractère récent interdit de tirer des conclusions définitives. Il nous semble néanmoins possible de dégager quelques premiers enseignements qui pourront être confrontés à l'évolution des cas étudiés et à l'étude d'autres expériences similaires.

7.3.1 Des stratégies contraintes.

Le premier enseignement est que dans les deux cas, le choix effectué de transformer le logiciel en logiciel libre résulte d'un échec commercial : échec du projet Quantum et tournant vers les services dans le cas de Matra Datavision, échec pour transformer Code Aster en un produit commercialisé dans le cas de EDF. Toutefois ces échecs commerciaux ne remettent pas en cause les qualités intrinsèques de ces logiciels. Comme nous l'avons montré, dans l'économie du logiciel l'importance des rendements croissants d'adoption et notamment des externalités de réseaux explique qu'une solution qui n'est pas (ou plus) techniquement optimale peut rester dominante. Cette particularité de l'économie du logiciel n'est pas toujours bien comprise : par exemple, le nouveau directeur de la R&D d'EDF à qui il était exposé le projet concernant Code Aster avouait son incompréhension devant l'affirmation qu'il s'agissait d'un «objet de valeur mais que l'on ne pouvait pas faire de business avec».

Par contre, il est impératif pour qu'une stratégie de transformation en logiciel libre ait des chances de succès que le logiciel soit de qualité : «le produit doit être bon, utile, fiable et de qualité» affirme Hugues Rougier (PDG de Matra Datavision). Le logiciel peut avoir du mal à se valoriser mais pour des raisons qui ne renvoient pas à des insuffisances techniques du produit. Dans une situation où un logiciel est confronté à une entreprise dominante qui détient la majorité du marché, sa transformation en logiciel libre apparaît comme une «arme contre les rentes de situation», «une démarche du faible au fort» (idem).

7.3.2 Des logiciels pour informaticiens : un modèle en couches.

Le deuxième enseignement est que le logiciel libéré doit être un logiciel utilisé par des informaticiens. En effet, la caractéristique distinctive d'un logiciel libre est que son code source est librement accessible, la gratuité du logiciel n'étant qu'une conséquence inévitable de cet état de fait. Mais comme l'indique Hugues Rougier, «l'accès aux sources doit servir à quelque chose». Or cet accès n'intéresse que les informaticiens, du moins directement. Pour un utilisateur non informaticien la connaissance du code source, pour ne pas parler de son éventuelle modification, ne présente aucune utilité si ce n'est que cette possibilité peut être éventuellement un signal de qualité.

Le type de logiciel où l'approche du logiciel libre peut être intéressante concerne les logiciels qui sont des outils utilisés par des informaticiens, soit pour effectuer des tâches comme la simulation numérique où l'ouverture du code facilite les adaptations à des besoins particuliers, soit pour réaliser des développements logiciels spécifiques comme dans le cas d'une plate-forme de développement. On peut ajouter que ces activités ne doivent pas être le cœur de métier de l'entreprise : par exemple, les entreprises qui utilisent un outil logiciel de simulation numérique dont le code est public veillent jalousement à la confidentialité et au caractère privé des données sur lesquelles reposent leurs simulations, qui constituent un aspect important de leur compétitivité.

La proportion des logiciels utilisés par des informaticiens est appelée à s'étendre avec l'extension du «modèle en couches» : celui-ci consiste en la poursuite de l'allongement du détour de production pour arriver

à des logiciels exploités par les utilisateurs finaux. De plus en plus les logiciels sont construits avec d'autres logiciels, eux-mêmes réalisés grâce à d'autres logiciels La forme extrême de ce processus est la production de logiciels à partir de composants logiciels, que facilitent les conceptions de la programmation orientée objets en forte croissance. Cette direction correspond à un des enjeux majeurs de l'économie du logiciel. Face à l'extension permanente de l'utilisation de logiciels et de la diversité des besoins, le développement de logiciels sur mesure à partir de composants standardisés (deuxième modalité du monde de production flexible) apparaît comme une solution permettant de concilier une productivité et une fiabilité relativement élevées (par la réutilisation de composants testés) tout en pouvant ajuster le logiciel développé aux besoins précis des utilisateurs. Mais le développement de cette forme de production particulièrement prometteuse ne va pas de soi. Pour être pleinement efficace, il faut pouvoir utiliser des composants logiciels choisis parmi l'ensemble de ceux qui ont été développés par la communauté informatique, et ne pas se limiter à ceux qui ont été produits en interne. Le problème est que l'utilisation des composants logiciels nécessite l'accès direct au code source de ces composants pour résoudre les problèmes d'interopérabilité, de fiabilité et de pérennité. L'existence de ces composants sous forme de composants logiciels libres peut donc constituer une solution intéressante, l'entreprise qui libère une «couche» de briques logicielles pouvant financer cette activité par l'ajout d'une dernière «couche» (logicielle ou de services) où elle escompte être plus compétitive.

7.3.3 La recherche de l'imposition d'un standard

Cette évolution renforce l'importance des questions de standardisation qui ont toujours été des questions décisives pour les logiciels comme plus généralement pour l'ensemble de l'informatique vu l'intensité des interrelations technologiques entre les différents composants logiciels et matériels des systèmes informatiques. De ce point de vue, la transformation en logiciel libre peut être un moyen efficace pour imposer un standard, pour devenir la référence incontournable dans un domaine. Par exemple, le formidable développement d'Internet et du World-Wide-Web est dû à la réussite des processus de standardisation sur des standards ouverts qui s'appuient sur des logiciels libres. Réciproquement, l'existence d'Internet joue un rôle majeur dans la diffusion des logiciels libres comme dans la participation à leur amélioration et à leur développement.

Dans les exemples étudiés, l'espoir que des standards se créeront autour des logiciels libérés est une préoccupation majeure. Ceci explique la volonté d'être le premier dans le secteur considéré à effectuer cette démarche et le choix de certaines décisions, en termes de licence par exemple, pour bénéficier de la reconnaissance de la communauté du logiciel libre et accélérer les processus de standardisation. La réussite de ces processus conditionne l'extension d'un réseau soit pour améliorer le produit (EDF), soit pour commercialiser des services (Open Cascade). En même temps, si l'efficacité de cette démarche provient de son originalité, cela rend plus hypothétique d'éventuelles décisions de libérer d'autres logiciels dans des secteurs où existent déjà des logiciels libres. Par contre, on peut penser que les interconnexions entre les différents domaines peuvent inciter des acteurs proches d'un domaine où existe un logiciel libre à libérer leur logiciel. Ce n'est pas par hasard que les deux expériences d'entreprises industrielles ayant fait ce choix se situent dans des domaines connexes. Hugues Rougier estime que par un phénomène de contagion progressive vont s'étendre les zones où existent des logiciels libres sur la «carte» des logiciels.

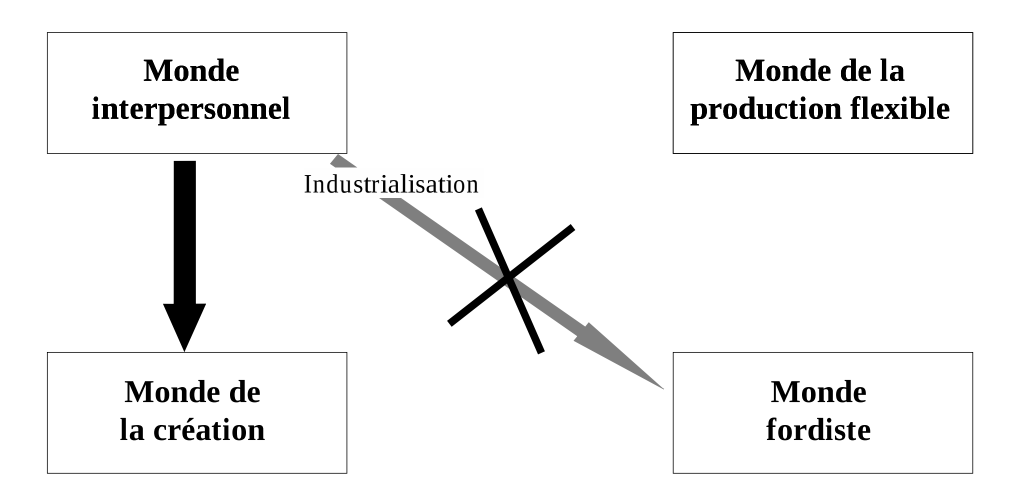
7.3.4 Des succès qui ne doivent pas masquer les difficultés pour s'adapter à des modèles économiques inédits.

Le succès principal concerne l'amélioration de la qualité du logiciel libéré qui se diffuse rapidement, ce qui contribue en retour à améliorer ses qualités. Au delà de la gratuité qui n'est qu'une conséquence, la caractéristique principale d'un logiciel libre est que l'on dévoile son contenu, que ce n'est plus une «boîte noire» : on peut voir comment il est architecturé, écrit, percevoir ses défauts, localiser précisément les dysfonctionnements alors qu'il est très difficile d'imputer à un logiciel particulier les problèmes rencontrés dans l'utilisation

d'un système informatique global sans disposer du code source des logiciels qui interagissent. C'est ce qui explique qu'une entreprise ne peut libérer que des logiciels de qualité, et que dans les deux cas étudiés d'importants investissements ont été consentis pour améliorer encore le logiciel avant de publier son code source. En effet, de telles décisions peuvent affecter l'image de l'entreprise au delà du seul logiciel libéré : c'est ce qui explique que certaines entreprises, comme Thomson, qui disposent d'un important portefeuille de logiciels qui ne sont plus valorisés sont rétives, non pas à fournir gratuitement ces logiciels, mais à rendre publics leurs codes sources. Par contre, un logiciel de qualité qui est libéré voit sa diffusion augmenter et bénéficie des contributions de ses utilisateurs à la poursuite de son amélioration.

Les deux cas étudiés confirment ce raisonnement. Mais ils montrent également les difficultés rencontrées pour adopter des modèles économiques radicalement nouveaux. Avant d'examiner ces difficultés, il est nécessaire de spécifier les deux modèles assez différents identifiés, dont on peut penser qu'ils ont une portée plus générale pour analyser d'autres expériences de transformation en logiciels libres.

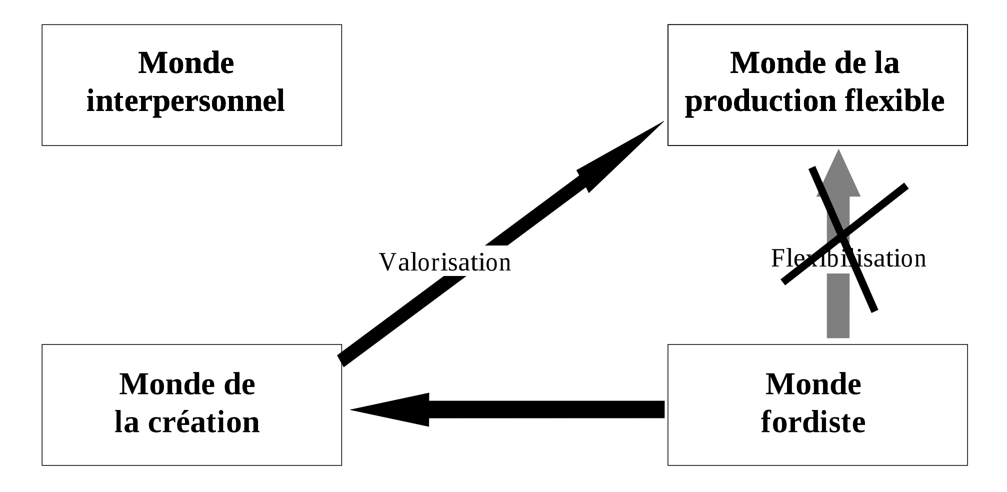
***Figure 7.2* —** Un premier modèle : pérenniser un produit stratégique en le libérant.



Le premier de ces modèles, illustré par l'expérience d'EDF repose sur la volonté de pérenniser un produit jugé stratégique. Dans la représentation que nous avons fait des évolutions dynamiques de l'économie du logiciel, ce modèle correspond à une entreprise située dans le monde de production interpersonnel (un logiciel développé pour répondre à des besoins internes) qui ne parvient pas à transformer son logiciel en progiciel commercial (échec de la trajectoire que nous avons qualifié d' «industrialisation»). La transformation en logiciel libre correspond sur le schéma à une nouvelle trajectoire qui part du monde interpersonnel vers le monde de la création, l'objectif étant de continuer à améliorer qualitativement le logiciel pour les besoins internes de l'entreprise grâce aux retours d'expériences d'utilisateurs plus nombreux et plus diversifiés tout en minimisant les coûts de développement par l'existence de contributeurs extérieurs. Par contre, l'objectif de l'entreprise n'est pas d'atteindre le monde de production flexible par la commercialisation de services et de développements sur mesure à partir du logiciel libéré (trajectoire appelée «valorisation»). Si une telle évolution est souhaitable pour assurer la pérennité du logiciel, elle sera l'œuvre d'acteurs extérieurs à l'entreprise (avec une aide au démarrage éventuelle de l'entreprise) sans que l'entreprise en retire un bénéfice commercial. Si dans le cas étudié, c'est le caractère stratégique du logiciel pour l'entreprise qui est mis en avant, on peut également concevoir qu'un tel modèle soit pertinent pour des logiciels jugés stratégiques au niveau d'un pays (par exemple, la recherche d'une indépendance technique pour la maîtrise des logiciels des couches basses des activités de défense) ce qui pose la question d'un financement public éventuel pour l'adoption de ce modèle.

Le deuxième modèle, correspondant à l'expérience d'Open Cascade, repose sur la vente de services divers liés au logiciel libéré, y compris des développements logiciels sur mesure. Ici l'objectif est de se positionner

Figure 7.3 **—** Un deuxième modèle : vente de services liés au logiciel libre.



dans le monde de production flexible considéré comme étant l'avenir d'une activité lucrative dans l'économie du logiciel. Le point de départ de cette évolution est une entreprise située dans le «monde fordiste» en tant qu'éditeur de progiciels. Face aux impossibilités pour développer ces nouvelles activités à partir d'un progiciel en situation d'échec commercial (difficultés pour s'insérer sur la trajectoire dite de «flexibilisation»), le choix est fait d'évoluer vers le monde de la création (transformation en logiciel libre) mais avec la volonté de profiter de ce positionnement pour que l'entreprise puisse atteindre le monde de production flexible en suivant la trajectoire dite de «valorisation». La transformation en logiciel libre est dans ce cas une démarche marketing particulière qui représente un investissement que l'entreprise espère rentabiliser par un avantage compétitif dans la prospection des clients et dans la qualité des solutions proposées sur le marché des services liés au logiciel libre. La pertinence de ce modèle implique qu'il existe un marché conséquent pour des services complémentaires au logiciel libéré : par exemple, Intelli CAD a tenté sans succès de suivre cette stratégie en libérant son logiciel Visio pour concurrencer Autodesk dans le domaine de la CAO d'entrée de gamme, mais s'est retrouvé dans une situation délicate en raison du manque de services réels proposés (Industries et Techniques, février 2001).

Par ailleurs la mutation d'entreprises vers ces modèles économiques novateurs présente des difficultés.

7.3.5 L'importance des changements culturels.

La première de ces difficultés réside dans l'importance des changements culturels que représente cette transformation pour l'entreprise. Ces changements culturels concernent tout d'abord la façon de produire du logiciel. Pour les développeurs, «cela leur fait quelque chose de publier leur source» explique Jean-Raymond Lévesque. Il existe une certaine crainte à dévoiler l'intimité de leur travail, à révéler publiquement la façon dont ils ont conçu et écrit le logiciel en offrant prise à une critique externe beaucoup plus importante : «est-ce que l'on va écrire de la même façon maintenant que c'est libre ?» s'interroge le responsable qualité de Code Aster. Pour des informaticiens d'entreprise, produire des logiciels libres constitue une rupture plus brutale que pour les informaticiens universitaires, qui ont joué un rôle important dans le développement des premiers logiciels libres, et qui sont habitués à une évaluation basée sur la publicité des travaux et fondée sur le jugement de leurs pairs. En même temps, l'efficacité des critiques et le niveau plus élevé des exigences qui en découlent, expliquent les qualités des logiciels libres par rapport à des logiciels non libres où le manque de transparence masque des défauts de conception difficilement identifiables à partir de la seule visibilité des effets d'un logiciel intégré dans un système informatique global.

Il existe également des changements culturels substantiels dans la nature de l'activité économique. Par exemple, dans le cas d'EDF la culture dominante est devenue la nécessité pour chaque collaborateur de créer de la valeur («vous n'êtes pas au CNRS !» est une critique fréquemment formulée), et il n'est pas toujours évident, pour les développeurs comme pour les managers, de s'insérer dans une activité qui apparaît désintéressée. De même dans le cas d'Open Cascade, Abou-Haidar explique : «lorsque nous leur avons annoncé que leurs travaux allaient être livrés gratuitement à tout le monde, c'est un peu comme si on leur arrachait un bras. Il fallait leur faire prendre conscience de partager leur travail. Cela a été difficile» (Linux +, septembre 2000). Inversement, l'adaptation n'est pas non plus évidente pour certains salariés dont le comportement était façonné par l'implication dans un département de R&D et qui doivent se reconvertir en prestataire de services commercialisés, comme dans le cas d'Open Cascade.

Dans tous les cas l'adhésion indispensable des intéressés à des changements de valeurs, les transformations d'identités professionnelles qui en résultent, sont des processus qui ne vont pas de soi et qui prennent du temps[8].

7.3.6 Au moins à court terme plus de bénéfices sociaux que privés.

On peut considérer que la transformation d'un logiciel en logiciel libre représente un bénéfice social plus ou moins important selon le succès du logiciel libéré, mais implique un éventuel manque à gagner (absence de redevance possible sur le logiciel) et surtout des investissements pour l'entreprise qui prend cette décision. La rentabilité de ces dépenses pour l'entreprise n'est pas toujours évidente et ne peut se manifester qu'à terme. Du reste, il apparaît assez clairement que les deux expériences étudiées ne peuvent se poursuivre que parce que les entités qui mènent ces projets sont adossées à de grands groupes (EDF, EADS) pour lesquels les dépenses supplémentaires à engager ou les pertes initiales ne représentent qu'une fraction dérisoire de leur chiffre d'affaires. Comme le souligne la Direction générale de l'Industrie, des Technologies de l'Information et des Postes, du Ministère de l'Économie, «une stratégie «open source», supposant un gros investissement, sans recettes découlant directement du produit développé, n'est évidemment pas exempte de risques pour son promoteur, alors même qu'elle se traduit par d'importantes retombées pour la communauté d'utilisateurs» («La R&D industrielle, une clé pour l'avenir : Six cas exemplaires d'entreprises», p. 79).

Une situation où le rendement social d'une initiative dépasse son rendement privé pose logiquement la question d'un financement public au moins partiel de ce type d'activités si l'on estime qu'il est souhaitable que ce type d'initiative ne reste pas marginal et puisse s'étendre à de nombreuses situations. Un financement public peut constituer un «coup de pouce» indispensable pour initialiser un projet, notamment quand il ne s'agit pas de libérer le code source d'un logiciel existant mais de construire dès le départ un logiciel libre : par exemple, des dentistes allemands souhaitaient développer sous forme de logiciel libre, un logiciel d'usinage des prothèses dentaires qui devait permettre à partir de capteurs numérique de réaliser l'usinage des prothèses directement dans le cabinet dentaire ; ce projet d'un coût de 5 millions de marks devait être financé par une souscription des dentistes et a échoué parce qu'il manquait 300.000 marks.

A contrario, le projet Salomé («Simulation numérique par Architecture Logicielle en Open source et à Méthodologie d'Évolution») représente «un cas très intéressant et sans doute encore rare, de mise en œuvre d'une véritable stratégie industrielle reposant sur le logiciel libre.» («La R&D industrielle, une clé pour l'avenir : Six cas exemplaires d'entreprises», rapport de la Direction générale de l'Industrie, des Technologies de l'Information et des Postes, p. 60). Ce projet se situe à l'interface entre la CAO et la simulation numérique, secteurs où se déroulent les deux expériences étudiées, ce qui confirme l'idée de contagion de proche en proche des domaines où existent des logiciels libres. L'idée poursuivie dans le projet SALOMÉ est de construire une plate-forme générique de liaison CAO-calcul pour la simulation numérique. Cette plate-forme

[8] Réagissant à une présentation dans le cadre du deuxième workshop, des représentants d'Alcatel et de Thomson ont confirmé la difficulté culturelle que représentait pour leurs développeurs de publier leur code source, de le faire partager à l'extérieur et à l'intérieur des entreprises. C'est, pour eux, le principal frein à la mise en libre de certains de leurs logiciels.

devrait comporter plusieurs composants modulaires et configurables, adaptables à toutes sortes de domaines technologiques. Le projet part d'un constat : la complexité des produits technologiques modernes et la fiabilité exigée d'eux imposent des simulations de plus en plus fines de leur comportement pendant la phase de développement. Certains logiciels de CAO proposent des modules de simulation, mais ils ne sont généralement pas suffisants et ne couvrent pas tous les domaines. La plupart des industriels utilisent donc des logiciels de calcul, ou solveurs, spécialisés, développés pour eux ou achetés sur le marché. Mais la liaison entre la CAO et ces outils de simulation est rien moins qu'aisée. Les informations introduites dans les solveurs doivent être présentées de façon très normalisée, d'où le fait que la «mise en données» doive souvent être effectuée «à la main». Cette phase manuelle, qui sert à faire le lien entre des outils informatiques d'une puissance colossale, peut dans certains cas représenter jusqu'à 50 % des temps d'étude.

Le projet SALOMÉ est un projet du RNTL (Réseau National des Technologies Logicielles constitué sous l'impulsion des Ministères de l'Industrie et de la Recherche) qui le finance à hauteur de 40 %. Les développements effectués dans le cadre de ce projet sont sous forme de logiciel libre avec une licence fondée sur trois principes : l'accès aux sources des composants de SALOMÉ est libre, toute correction d'anomalie réalisée dans SALOMÉ doit obligatoirement être publiée en open source, tout utilisateur de SALOMÉ est libre du choix de la licence d'exploitation relative aux composants complémentaires qu'il a développés. Le choix d'un modèle de diffusion gratuite, avec support technique et libre accès au code source, dans le but de constituer une communauté d'utilisateurs, clients potentiels de services associés, s'inscrit dans la stratégie du RNTL de promouvoir des logiciels libres «dans les domaines où l'industrie française est peu présente, [et où] le logiciel libre constitue un moyen efficace pour élargir les espaces d'exploitation d'innovations technologiques ou accéder à des positions compétitives». À première vue, il pouvait sembler difficile de trouver des partenaires prêts à engager des dépenses importantes pour développer un logiciel destiné à être distribué gratuitement. Pourtant, dans le projet SALOMÉ, cette idée a été partagée avec enthousiasme par neuf partenaires, éminents représentants de l'industrie (Open CASCADE, le Bureau Véritas, Principia, Cedrat, EDF et le centre commun de recherche d'EADS) et de la recherche publique (CEA, l'Institut National Polytechnique de Grenoble et l'Université Pierre et Marie Curie Paris VI). La plate-forme logicielle doit être disponible en septembre 2002 et il existe une convention pour continuer des développements, toujours sous forme de logiciels libres, après 2002.

Cet engouement est lié à la difficulté du projet. En effet, les outils logiciels développés doivent pouvoir accepter des informations propres à de nombreux domaines technologiques comme la mécanique, l'hydraulique, l'électromagnétisme, etc. Ces domaines sont si nombreux et si spécifiques que le travail de R&D nécessaire au développement de tels outils peut paraître hors de portée de n'importe quel éditeur ce qui explique que le créneau visé par SALOMÉ n'était pas encore occupé, alors que le besoin est manifeste («l'arlésienne des labos» selon un spécialiste du secteur). La réussite d'un tel projet suppose qu'il devienne un standard de facto et donc qu'il conquiert le terrain suffisamment vite, afin d'imposer une technologie qui deviendra une référence indispensable pour une grande partie des utilisateurs. Sinon, si le projet de standard industriel est supplanté par un projet concurrent, les travaux effectués pour le mettre au point l'auront été en pure perte. De plus, l'aspect générique de la plate-forme SALOMÉ est capital pour permettre de réduire les temps de formation à l'utilisation des outils mis en œuvre. Pour les utilisateurs, la possibilité d'étudier des phénomènes couplés plus aisément contribuera à réduire les coûts et la durée des études, et à augmenter la qualité des simulations, en favorisant l'interopérabilité entre les codes de simulation.

En plus de l'importance du soutien public, un facteur important de réussite de ce projet, est le fait qu'il répond à des attentes complémentaires des différents participants et qu'il correspond ainsi à une combinaison des deux modèles exposés précédemment. Pour les industriels, il s'agit de mieux rentabiliser les investissements souvent très importants effectués pour développer des codes de calculs de simulation numérique dans les différents domaines qui constituent le cœur de leur métier, en disposant d'outils d'intégration dans une plate-forme globale avec les logiciels de CAO. De plus les choix de codes calcul effectués sont souvent difficilement réversibles car les vérifications des simulations reposent fréquemment sur des itérations par rapport

à des codes calcul existant déjà testés. Pour ces futurs utilisateurs de la plate-forme logicielle, la participation de leurs équipes aux efforts communs de développement relève d'une forme de veille technologique, en contribuant à l'élaboration d'outils nécessaires dans le cadre d'un travail de R&D relevant de leur secteur d'industrie, que ce soit le pétrole, l'aéronautique ou le génie électrique. Par contre, pour Open Cascade, Salomé représente une extension naturelle de sa plate-forme logicielle et elle espère rentabiliser sa participation au projet par la vente de services liés : par exemple, le CEA a d'ores et déjà conclu des contrats avec Open Cascade pour l'intégration de Salomé.

7.4 Conclusion

L'analyse de l'économie du logiciel comme étant fondamentalement marquée par *la coexistence permanente de plusieurs mondes de production, qui reposent sur l'existence de conventions différentes, centrées sur un type de produit, et constituant des réponses différenciées aux questions critiques de l'économie du logiciel*, permet de mettre en évidence les fondements de la diversité observée dans l'économie du logiciel. L'étude des évolutions dynamiques à l'œuvre montre que, si l'importance respective et les formes concrètes des différents mondes de production évoluent, l'existence de plusieurs mondes de production est un phénomène durable. En particulier, l'originalité, que constituent la production de logiciels libres et son articulation avec des formes marchandes de production, est un phénomène positif pour l'ensemble de l'économie du logiciel, en permettant une certaine diversité des logiciels produits et en favorisant l'avènement d'un véritable «sur mesure de masse» dans la production des logiciels. Par contre, il n'est pas évident que la création ex nihilo de logiciels qui sont d'emblée des logiciels libres, comme l'ont été la plupart des logiciels libres actuels, soit suffisante pour répondre à l'importance et à la diversité des besoins. De ce point de vue, l'autre voie pour que des logiciels libres existent, à savoir transformer des logiciels au départ «privés» en logiciels libres, peut être une solution complémentaire intéressante. Quelques entreprises pionnières se sont engagées récemment dans cette voie. Les nouveaux modèles économiques vers lesquelles elles évoluent sont dotés d'une indéniable cohérence, mais il est évidemment trop tôt et les exemples sont trop peu nombreux pour pouvoir conclure sur la réussite pratique de telles stratégies. Il apparaît en tout cas clairement que de telles évolutions prennent du temps et nécessitent des moyens significatifs avant que d'éventuels résultats positifs se fassent sentir. Il faudra suivre avec attention la poursuite de ces expériences car leur succès (ou leur échec) auront un impact non négligeable sur l'extension de ces stratégies à d'autres entreprises intéressées, mais qui veulent vérifier la crédibilité de ces nouveaux modèles avant de tenter de les adopter. En attendant, la réalisation de l'objectif souhaitable que des logiciels libres apparaissent dans la plupart des domaines de l'économie du logiciel - notamment dans les secteurs dominés par des entreprises américaines et/ou dans les secteurs où il n'existe pas de logiciels «privés» répondant de façon satisfaisante à l'ensemble des besoins (comme le créneau couvert par Salomé) - impliquera nécessairement la poursuite et l'accentuation des impulsions que peuvent apporter les pouvoirs publics.

La licence : un outil stratégique pour les éditeurs de logiciels.

LAURE MUSELLI,
Université de Paris XIII, laure_muselli@yahoo.fr.

Avec la montée en puissance du mouvement des logiciels libres et devant la multiplicité des licences informatiques utilisées par les entreprises éditrices de logiciels, on peut être amené à s'interroger sur les raisons qui orientent le choix d'une entreprise pour tel type de licence ou tel autre. étant données les particularités de l'industrie des logiciels, apparentée à une industrie de réseau où la guerre des standards fait rage, on peut se demander si les licences ne constitueraient pas un outil stratégique puissant, construit de façon très fine pour «coller» au mieux aux objectifs de l'entreprise. Car une des particularités de l'industrie du logiciel réside dans le fait que l'entreprise désirant tirer des bénéfices de l'exploitation d'un logiciel qu'elle a conçu ne se place pas dans une logique marchande classique. En effet, elle ne vend pas ses logiciels comme s'ils étaient des produits, mais passe des accords de licence avec tout individu désirant les utiliser, en les assimilant ainsi à une technologie : plutôt qu'un contrat de vente, c'est une concession de droits d'utilisation de sa technologie que l'entreprise pratique auprès des utilisateurs. La licence informatique, «contrat d'utilisation» avec l'utilisateur, est ce qui définit le droit de ce dernier à utiliser le logiciel, mais aussi les obligations plus ou moins restrictives qui accompagnent cette utilisation. Les entreprises peuvent, par exemple choisir une stratégie permettant de valoriser au maximum les droits d'auteur possédés en imposant des redevances élevées et des conditions d'utilisation strictes, alors que d'autres vont miser sur d'autres aspects stratégiques, comme la standardisation par exemple.

Cet article vise à montrer que l'on peut relier directement certains types de clauses à des objectifs stratégiques précis, sans toutefois oublier que le caractère statique d'une licence va à l'encontre du caractère évolutif d'une stratégie.

Dans une première partie, nous passerons en revue les caractéristiques des industries de réseau et nous présenterons les stratégies généralement mises en œuvre pour tirer partie de ces caractéristiques particulières. Après avoir dressé un tableau des nombreuses licences de logiciels existantes, nous établirons une typologie des stratégies susceptibles d'être servies par ces licences. Nous proposerons enfin, dans la troisième partie, une méthodologie permettant de relier ces différentes stratégies aux clauses de licences, que nous appliquerons à des cas concrets d'éditeurs de logiciels tels que Troll Tech ou Sun Microsystems.

8.1 Caractéristiques de l'industrie des logiciels : rendements croissants et standardisation.

Afin de répondre aux questions que nous allons aborder, il est nécessaire de comprendre qu'un logiciel est généralement composé de plusieurs modules, qui en font un produit enrichissable, modifiable et évolutif. Ceci nous amène à classer les différents types de logiciels en deux catégories en fonction de leur nature.

D'une part ce que nous appellerons les «technologies logicielles», vaste catégorie dans laquelle entrent les systèmes d'exploitation, les langages informatiques et les bibliothèques. Les logiciels classés dans cette catégorie sont destinés à être utilisés pour élaborer d'autres produits logiciels (comme dans le cas des langages ou des bibliothèques) ou pour les faire fonctionner (c'est le cas des systèmes d'exploitation, indispensables pour faire fonctionner des logiciels d'application sur un ordinateur), ce qui les apparente plutôt à des technologies logicielles, par analogie avec les technologies industrielles. En effet, comme une technologie industrielle n'a comme seul but que la fabrication de produits finals ou la gestion de cette production, les technologies logicielles n'ont d'intérêt que la gestion ou la production de produits logiciels dérivés.

D'autre part les logiciels d'application : ce sont des programmes utilisés comme des produits finals par les utilisateurs, et dérivés de technologies logicielles et autres logiciels à partir desquels ils sont conçus. Ce

sont par exemple les logiciels de bureautique comme le traitement de texte, des logiciels de calcul, de création multimédia, ou de mise en page. Ce type de logiciel peut être assimilé à un produit classique issu d'une technologie industrielle.

Dans cet article, nous nous focaliserons sur la catégorie des technologies logicielles, car leur caractère évolutif et modifiable, que ne possèdent pas les produits logiciels, en font un enjeu important pour les entreprises éditrices, qui voient en elles un moyen de contrôler des marchés dérivés si elles réussissent à imposer leur propre standard. On comprend ainsi mieux leur intérêt à déployer des stratégies agressives visant à tirer parti des mécanismes naturels jouant dans les industries de réseau, dont l'industrie du logiciel fait partie.

8.1.1 Les caractéristiques d'une industrie de réseau.

Si les préoccupations stratégiques des entreprises éditrices de ce type de logiciels apparaissent aussi importantes, c'est qu'elles sont à rapprocher des particularités de ce secteur, qui s'apparente à une industrie de réseau : d'une part la présence de fortes externalités de réseau induisant un mécanisme d'effet feed-back, et d'autre part une importante possibilité de verrouillage des utilisateurs[1]

De fortes externalités de réseau induisant un mécanisme d'effet feed-back.

Comme l'a déjà signalé Marie Coris dans cet ouvrage, l'informatique est une industrie dans laquelle la valeur d'un logiciel est bien souvent fonction du nombre d'autres agents l'utilisant déjà (c'est ce qu'on appelle le réseau d'utilisateur). Ainsi, l'entrée d'un nouveau «membre» dans le réseau fera bénéficier tous les autres d'une valeur supplémentaire, créant une externalité de réseau. Et, à qualité de produit égale, un nouvel arrivant ayant le choix entre deux réseaux, aura tendance à choisir le plus vaste. Sur le marché des technologies logicielles, où le réseau est constitué par les utilisateurs d'une même technologie (langage, système d'exploitation, bibliothèque, etc.), ce principe est largement vérifié dans la mesure où l'utilisateur d'une technologie isolée ne pourra ni communiquer avec d'autres agents, ni proposer des produits dérivés interopérables. Cela produit un effet de «feed-back» : plus une technologie est appréciée et adoptée par des agents, plus les autres anticiperont qu'elle va devenir le standard et l'adopteront pour profiter des externalités de réseau évoquées plus haut. Un cercle vertueux se forme alors, et la technologie s'impose comme standard sur le marché. A contrario, lorsqu'une technologie est en perte de vitesse, sa chute va aller en s'accélérant. En effet, si la technologie est peu adoptée par les premiers acheteurs, les suivants anticiperont qu'elle ne deviendra pas le standard et risque même de disparaître. Par conséquent ils ne l'adopteront pas, du fait de l'absence d'externalités de réseau. Cette fois, c'est un cercle vicieux qui se forme, entraînant la perte de la technologie dans la bataille de standard.

Ainsi, l'effet feed-back est la conséquence du désir des acheteurs de profiter des externalités de réseau, c'est-à-dire d'appartenir au réseau le plus large et d'acquérir la technologie qui finira par s'imposer comme standard de facto. C'est souvent la technologie atteignant la première une masse critique d'utilisateurs qui s'imposera, alors que la ou les autres disparaîtront.

Des possibilités de verrouillage.

La deuxième caractéristique des industries de réseau et particulièrement de l'industrie informatique est la présence de verrouillage des utilisateurs dans des situations où les coûts de changement sont trop importants

[1] Sur les caratéristiques de ces industries, on consultera, en plus des ouvrages déjà cités dans les articles précédents, Farrell [1989], Foray [1989], Garud et Kumaraswany [1993], David et Foray [1995], Dalle [1995], Jullien et Zimmermann [2002].

et les obligent à conserver les technologies adoptées. Ce verrouillage peut intervenir sur *les équipements durables, qui imposent des coûts de changement croissants avec le temps, notamment à cause des équipements complémentaires compatibles acquis au fil du temps* (c'est notamment le cas pour Microsoft, qui fait en sorte de défavoriser la compatibilité de logiciels d'application autres que les siens avec son propre système d'exploitation Windows) ; sur *l'apprentissage, qui, en cas de changement génère des coûts de formation ou des pertes de productivité* (comme le fait d'apprendre à utiliser Linux à la place de l'environnement Windows) ; enfin sur les *informations et bases de données, pour lesquelles un changement d'équipement occasionne des coûts de conversion des données aux nouveaux formats* (on peut citer comme exemple les formats de fichiers propres à chaque logiciel) Shapiro et Varian [2000].

Ces particularités favorisent l'établissement de standards garantissant à l'entreprise ayant réussi à les établir une domination du marché. On comprend alors mieux les efforts des entreprises du marché de l'édition de logiciels pour établir des stratégies visant à imposer coûte que coûte leurs technologies à l'ensemble des utilisateurs, sous peine d'être dominées par leurs concurrents. Compte tenu de la domination du marché que peut entraîner, pour une firme, l'établissement de sa technologie logicielle comme standard, les entreprises vont chercher à tirer parti du mécanisme de l'effet feed-back naturellement présent en tentant de l'amorcer rapidement et de l'amplifier. C'est ainsi qu'elles vont mettre en place des stratégies qui auront pour objectif la mise en place de standards de facto.

8.1.2 Les stratégies tirant profi t de l'effet feed-back.

Shapiro et Varian [2000] ont mis en évidence deux stratégies permettant de tirer profit, de façons différentes, de l'effet feed-back, qui sont le contrôle et l'ouverture.

Stratégie de contrôle.

Elle consiste, pour l'entreprise, à contrôler totalement la diffusion et l'évolution de sa technologie. Elle se traduit généralement par un contrôle de l'amélioration de la technologie, ainsi que du rythme de mise à disposition des versions supérieures (exemple Microsoft), par un contrôle de sa diffusion, l'entreprise restant la seule à pouvoir délivrer des licences d'utilisation, ou enfin par le contrôle de son éventuelle compatibilité avec les technologies d'autres firmes (en empêchant l'accès au code source par exemple ou en introduisant volontairement des incompatibilités avec les systèmes concurrents).

Une fois la technologie adoptée comme standard, les avantages de cette stratégie sont évidents : l'entreprise n'a aucune concession à faire en termes de compatibilité, car elle reste la seule maîtresse de ses choix. Sa position n'en est que renforcée et les concurrents sérieux ont peu de chance de résister et de l'inquiéter ; d'autre part, elle est également en mesure de fixer les prix et les conditions de ses licences à sa guise, les clients étant verrouillés ; enfin, elle n'est pas dans l'obligation d'améliorer ses produits constamment, du fait du verrouillage des clients et de l'absence de concurrents sérieux sur le marché.

Mais il faut convenir que la constitution de la base installée est difficile à réaliser. En effet, les clients potentiels peuvent se méfier de cette stratégie de contrôle susceptible de les mettre dans une situation de dépendance embarrassante par la suite et choisir une technologie concurrente. La masse critique est alors difficile à atteindre.

Stratégie d'ouverture.

La stratégie d'ouverture se place à l'opposé de la stratégie de contrôle, dans la mesure où elle mise sur l'abandon du contrôle de la technologie au profit de conditions incitatives plus favorables à son adoption par les utilisateurs. Cette stratégie peut elle-même être décomposée en deux sous-stratégies qui sont la normalisation et l'ouverture totale.

Normalisation.
La normalisation consiste à confier le choix d'un standard commun à un organisme de standardisation, tel que l'ISO ou l'ECMA (European Computer Manufacturer Association), dont le but sera de trouver un consensus entre tous les concurrents présents dans le comité. Il peut arriver que le choix de la technologie ait déjà été réalisé par les industriels avant l'intervention de l'organisme, mais que celui-ci ait un rôle de garant de l'ouverture du standard dans l'avenir : c'est lui qui dictera les évolutions futures de la technologie.

Une entreprise faisant en sorte que sa technologie devienne un standard «de jure» (norme) pourra atteindre une base installée relativement rapidement en évitant une bataille de standard coûteuse, et incertaine, puisque les utilisateurs auront une garantie quant à la pérennité et la compatibilité de cette technologie. Cependant, ce type de stratégie occasionne une perte de contrôle au profit de l'organisme de standardisation auquel seront confiés les choix des évolutions futures. Ainsi, la rente de monopole de l'entreprise acquise par l'entreprise sera moins élevée que dans le cas d'une stratégie de contrôle, dans la mesure où toutes les entreprises du marché seront en mesure d'utiliser, au même titre qu'elle la technologie standard. Cette situation de concurrence entraîne une impossibilité de «lock-in» des utilisateurs sur une offre et par conséquent des prix plus bas. D'autre part, les coûts de R&D risquent d'être accrus, dans le but de produire des produits de qualité, afin de conserver sa part de marché.

Ouverture totale.
C'est la situation où une entreprise décide de mettre sa technologie à disposition de tous, sans restriction, de façon à ce que n'importe quelle autre entreprise puisse fabriquer des produits compatibles. Cette stratégie mise sur l'annonce d'une **absence totale de «lock-in»** pour les utilisateurs, de façon à acquérir une base installée suffisante afin de déclencher rapidement un effet feed-back important. Les utilisateurs, mis en confiance par l'absence de perspective de «lock-in», vont adopter plus volontiers la technologie et permettre à l'entreprise l'acquisition d'une base installée conséquente.

L'inconvénient reste le fait que malgré l'adoption de sa technologie en tant que standard, l'entreprise ne possède plus réellement d'avantage par rapport à ses concurrents, capables au même titre qu'elle de fabriquer des produits compatibles. Cependant, lorsque l'issue de la bataille de standard semble incertaine, l'attrait d'un produit totalement ouvert reste alors une des seules chances pour l'entreprise d'imposer son standard, même si la perte totale de contrôle de la technologie, la firme ne peut envisager faire des bénéfices conséquents sur la technologie elle-même. Elle doit donc, pour tirer parti de cette stratégie, posséder les compétences et l'avance nécessaires pour proposer des produits complémentaires ou des services payants et se rémunérer ainsi.

Une fois le choix stratégique effectué, entre contrôle et ouverture, l'entreprise va disposer d'un grand nombre d'outils pour mener à bien sa stratégie d'imposition de standard. Dans la suite de cet article, nous allons plus particulièrement étudier l'un d'entre eux, la licence informatique, en essayant de montrer que loin d'être anodines, elle peut s'avérer être un puissant outil stratégique au service de l'entreprise.

8.2 Stratégies de licence, une typologie.

Avant de nous pencher plus précisément sur les différentes licences existantes, nous commencerons par rappeler quelques concepts indispensables à la compréhension de la typologie présentée.

8.2.1 Quelques concepts.

Le système de protection intellectuelle auquel sont soumis les logiciels est le droit d'auteur, ce qui signifie que toute entreprise mettant au point un logiciel bénéficie sur celui-ci d'un droit de propriété intellectuelle qui est le droit d'auteur. Les licences ne sont rien d'autres que la matérialisation de l'utilisation que va faire l'entreprise de ses droits de propriété intellectuelle sur ses logiciels[2].

Ainsi, l'entreprise désirant tirer des bénéfices de l'exploitation d'un logiciel qu'elle a conçu ne se place pas dans une logique marchande classique. En effet, elle ne vend pas ses logiciels comme s'ils étaient des produits, mais passe des accords de licence avec tout individu désirant les utiliser, en les assimilant ainsi à une technologie : plutôt qu'un contrat de vente, c'est un contrat de concession de droits d'utilisation de sa technologie que l'entreprise pratique auprès des utilisateurs. Cela donne à l'utilisateur le droit d'utiliser le logiciel, en lui imposant en contrepartie des obligations plus ou moins restrictives ainsi qu'une redevance plus ou moins élevée.

Des entreprises vont choisir une stratégie permettant de valoriser au maximum les droits d'auteur possédés en imposant des redevances élevées et des conditions d'utilisation strictes, alors que d'autres vont miser sur d'autres aspects stratégiques, comme la standardisation par exemple.

Par exemple, le Copyleft, concept créé par Richard Stallman, fondateur en 1980 de la Free Software Foundation, a pour but d'offrir un outil de protection au projet GNU, pour éviter aux logiciels «copyleftés» de «tomber aux mains» d'entreprises qui y attachaient des droits de protection assortis de licences d'exploitation propriétaires et restrictives[3]. Si le copyright, associé à une licence propriétaire, a souvent pour vocation de restreindre les libertés de l'utilisateur d'un logiciel, ce sont, par opposition, les clauses de la licence du copyleft qui sont, elles, présentes pour adjoindre des libertés au copyright. Ces clauses supplémentaires font donc en sorte de donner des libertés sur un logiciel «copyrighté», alors que celles d'une licence propriétaire adjointe au copyright autorisent la rente de monopole. Ce n'est donc pas le copyright, droit automatique donné à tout auteur, qui détermine la rente de monopole ou le caractère libre attachés au logiciel, mais bien la licence, selon que ses clauses limiteront les libertés ou, au contraire, les préserveront.

Ce principe de copyleft va être utilisé avec des clauses différentes selon les types de licence que nous allons passer en revue.

[2] Il s'agit ici d'étudier les conséquences stratégiques des clauses des licences. Mélanie Clément-Fontaine en fait l'analyse juridique dans la partie suivante. Ainsi, dans cet article, les termes «copyright» et «droit d'auteur» sont considérés comme synonymes. Sur ce sujet, on pourra consulter, en économie, Gallini et Winter [1985], Katz et Shapiro [1986], Bousquet et Wolkowicz [1994], Kotabe et al. [1996].

[3] Pratiquement, mettre un logiciel sous copyleft consiste en 2 étapes. Tout d'abord, le placer sous un copyright classique. Ensuite, coupler ce copyright à une licence, c'est-à-dire des clauses supplémentaires relatives aux conditions de distribution du logiciel. Ces conditions, qui peuvent varier selon la licence utilisée, stipulent généralement que tout utilisateur a la permission d'exécuter, de modifi er ou de distribuer librement le logiciel, dans la mesure où il transmettra ces mêmes droits aux utilisateurs de toutes les copies ou travaux dérivés dudit logiciel, sans y ajouter quelque restriction que ce soit. Cela signifi e que tout utilisateur ayant modifi é le code source d'un logiciel sous copyleft est obligé de faire du produit obtenu un logiciel placé sous licence de type libre. Afi n de protéger les logiciels libres, l'utilisation du copyright est la meilleure solution qui ait été trouvée. En effet, si l'auteur décide d'abandonner son copyright, pour permettre une diffusion, le programme tombe dans le domaine public, ce qui autorise tout individu ou entreprise , d'une part à continuer à diffuser le logiciel, mais aussi à placer des droits dessus pour éventuellement le transformer en logiciel propriétaire, et en restreindre ainsi l'utilisation.

8.2.2 De très nombreuses licences.

Les licences dites «de domaine public».

Les produits sous contrôle de ce type de licence ne sont pas exempts de tout droit de propriété intellectuelle car ils sont protégés par un copyright. Cependant, les clauses ajoutées attribuent aux utilisateurs les droits illimités de l'**utiliser**, le **copier**, le **modifier**, le **fusionner** [avec d'autres logiciels], le **publier**, le **distribuer**, le **sous-licencier** et/ou en **vendre des copies**, permettre aux personnes auxquelles elle les distribue de faire de même.

Les licences libres.

Deux associations, la FSF et l'OSI, ont établi une liste de critères permettant de définir le concept de Liberté et de déterminer l'appartenance ou non d'une licence à cette catégorie. Même si les critères mis en avant par ces deux associations diffèrent quelques peu, il apparaît que leurs conclusions tendent à se rejoindre. Pour la FSF, la signification du mot «libre» peut se résumer en 3 points : la liberté d'utiliser le logiciel, c'est-à-dire d'exécuter le programme, la liberté d'accéder au code source, quelle qu'en soit la finalité (étudier le fonctionnement du programme, adapter le logiciel à ses besoins personnels, améliorer le programme en en corrigeant les erreurs), la liberté de distribuer des copies du logiciel, par quelque voie que ce soit, **à titre gratuit ou pas.** Ainsi, tout logiciel régi par une licence stipulant l'exclusion d'une de ces libertés ne peut être qualifié de «libre». Le mot liberté sous-entend ici qu'il n'est nullement nécessaire de demander d'autorisation pour effectuer une de ces actions : l'utilisateur souscrit automatiquement à la licence à partir du moment où il en effectue une. D'autre part, un utilisateur ne peut, quelles que soient les circonstances, placer un produit dérivé de ce type de logiciel sous une licence différente : tous les produits dérivés doivent rester libres.

Les licences propriétaires.

Ce type de licence est classiquement utilisé par une majorité d'éditeurs de logiciels. Les caractéristiques de ce type de licence sont les suivantes : une utilisation soumise à redevance, et l'interdiction de copier le logiciel, d'accéder au code-source, de modifier le logiciel et de le distribuer à qui que ce soit. Contrairement aux licences libres, le copyright est cette fois assorti de clauses restrictives limitant la diffusion et la modification du logiciel.

Les licences hybrides.

Ce type de licence combine au copyright à la fois des clauses empruntées aux licences libres et d'autres clauses empruntées aux logiciels propriétaires. En conséquence, ce type de licence ne peut être totalement assimilé ni à du libre car il ne remplit pas toutes les conditions énoncées par les organisations certificatrices, ni à du propriétaire.

On peut donner comme exemple la licence SCSL de Sun Microsystems, appliquée au langage de programmation Java. Cette licence comporte des clauses compatibles avec du libre comme celles relatives à l'utilisation libre et à la copie libre, ainsi qu'à la disponibilité du code source, mais en revanche, les clauses concernant la distribution ne satisfont pas aux critères de liberté : la distribution par des tiers de produits dérivés de ceux protégés par la SCSL reste libre si elle est faite gratuitement, mais elle ne l'est plus du tout dès qu'un agent désire tirer un profit de cette activité. Dans ce dernier cas, Sun Microsystems impose le versement de royalties, mais s'octroie également un pouvoir de certification des produits dérivés, sans laquelle l'entreprise n'a pas le droit de distribuer son produit.

Les raisons des choix de ce type de licence par les entreprises restent floues, et celles-ci ne se privent généralement pas pour communiquer sur le caractère libre de leurs produits, qui sont pourtant soumis à des licences hybrides. Interrogées, elles critiquent notamment l'effet viral de la GPL.

Il apparaît que les types de licences sont nombreux, chaque catégorie possédant encore en son sein des licences différentes, et certaines entreprises n'hésitant pas à en créer une propre. Devant une telle multitude de licences, on peut se demander quels sont les critères qui peuvent guider les choix des entreprises vers des licences de domaine public, libres, hybrides ou propriétaires, si le choix de chaque clause est anodin, ou si au contraire, il est réalisé dans un but stratégique précis. Pour répondre à cette question, il est intéressant de se pencher sur les enseignements stratégiques tirés de l'étude des accords de licence technologique.

8.2.3 Stratégies de licence. Une proposition de typologie.

Afin de montrer que les licences sont conçues ou choisies dans des buts stratégiques précis, il apparaît nécessaire de définir tout d'abord quels types de stratégies sont susceptibles d'être atteints à l'aide de l'outil que représentent les licences. Pour ce faire, il semble pertinent de se baser sur les travaux de Bessy et Brousseau [1998] qui, dans leur article, se sont attachés à montrer que les accords de licence de technologie peuvent être mis en place dans des buts stratégiques extrêmement différents les uns des autres[4]. Ces auteurs mettent en évidence quatre facteurs explicatifs des types de stratégies d'entreprise auxquelles sont liés les différents types d'accords de licence technologique existants : les **politiques des entreprises en matière de propriété intellectuelle**, la **dynamique du processus d'innovation**, le **degré de dépendance mutuelle des membres de l'industrie** et les **formes organisationnelles alternatives à l'accord de licence.**. Et les caractéristiques de ces accords de licence vont varier en fonction de la stratégie choisie par l'entreprise.

En nous inspirant de ce travail, nous aboutissons, dans le cas du logiciel, à quatre facteurs explicatifs que nous nommerons valorisation patrimoniale, contrôle des firmes concurrentes, création de coopération (source d'éventuelles économies de R&D), établissement de standards.

Valorisation patrimoniale.

Le premier type de stratégie, sans doute le plus immédiatement perceptible, est la mis en évidence est la valorisation patrimoniale des connaissances de l'entreprise, à travers ses titres de propriété intellectuelle. La firme détentrice d'un droit de propriété intellectuelle sur une technologie possède une rente de monopole du fait de son droit exclusif à exploiter cette technologie. La redevance reçue dans le cadre d'un accord de licence est donc une forme de compensation de la perte d'une partie de la rente de monopole lorsque l'entreprise détentrice cède le droit d'exploiter la technologie au licencié. La stratégie de valorisation patrimoniale consiste, pour le licencieur, à générer des revenus grâce à l'exploitation directe de ses droits de propriété intellectuelle, en passant des accords de licence avec d'autres entreprises désireuses d'utiliser les technologies produites. Les

[4]On pourrait remarquer que les entreprises du secteur informatique utilisent non pas des accords de licence de technologie mais des licences informatiques sur leurs produits et qu'il y a des différences significatives. Premièrement, les accords de licence de technologie se font sur un ou plusieurs brevets détenus par une entreprise. En ce qui concerne les logiciels, la licence concerne le droit d'auteur, puisque c'est le mode de protection légal des logiciels. Deuxièmement, les accords de licence technologique se concluent uniquement entre entreprises, l'une étant propriétaire d'un brevet, l'autre souhaitant utiliser la technologie. Dans le cas des licences de logiciels, le licencié peut être une entreprise nécessitant la technologie pour développer d'autres produits informatiques, mais également un utilisateur final, entreprise ou particulier, utilisant le logiciel comme un produit fini : ceci n'est jamais le cas dans les accords de licence technologique. Cependant les similitudes entre celles-ci, et, principalement dans leur principe, identique, qui consiste à transférer des droits d'exploitation, sous certaines conditions, nous permettent de nous appuyer sur cette analogie.

redevances ainsi issues des contrats de licence peuvent alors notamment servir à financer de nouvelles activités de R&D. Dans ce cas, les droits de propriété ne représentent que des actifs permettant de générer un revenu.

Ce type de politique est évidemment très courant dans le secteur de l'édition de logiciels. En effet, les licences dites «propriétaires» associées à des logiciels commercialisés par bon nombre d'éditeurs sont utilisées dans un seul but de valorisation. Les conditions de l'accord de licence informatique pourront varier en fonction de l'importance du pouvoir de l'éditeur sur le marché : plus celui-ci se rapprochera d'une situation de monopole, plus la licence pourra imposer des conditions d'utilisation du logiciel restrictives et une redevance élevée. Ainsi, dans le cas de Microsoft, la firme se trouvant dans une situation de quasi-monopole sur certain de ses produits, les clauses de la licence peuvent être très restrictives et la redevance très élevée, aucun produit concurrent équivalent (pas seulement du point de vue de la qualité mais aussi d'autres comme la compatibilité) n'étant présent sur le marché.

Contrôle des firmes concurrentes.

La deuxième perspective consiste, pour l'entreprise, à considérer les accords de licence comme des *moyens de positionnement de l'entreprise sur le marché et le contrôle des autres firmes*, et ceci de plusieurs façons : en fixant au licencié une redevance plus ou moins importante afin de contrôler ses coûts, en lui imposant des clauses de *grant-back* (*obligation de rétrocéder des droits d'exploitation au concédant pour les innovations qu'il dérive de la technologie initiale*) afin de contrôler ses progrès technologiques, en fixant des conditions permettant de dissuader des firmes menaçantes d'entrer sur le marché ou en en excluant les entreprises rivales. Ce type de stratégie, que nous nommerons par la suite «stratégie de contrôle des firmes concurrentes», est pratiqué dans le cas des licences informatiques propriétaires, notamment par le biais de certaines clauses restrictives. On peut citer entre autres celles interdisant le reverse-engineering. Ces clauses permettent à l'éditeur de contrôler les progrès technologiques de ses concurrents en ne leur donnant pas la possibilité de créer des produits dérivés ou compatibles, et en se réservant exclusivement cette possibilité.

En ce qui concerne la clause de *grant-back,* celle-ci est présente dans certaines licences que nous qualifierons d'hybrides, et qui combinent des caractéristiques d'une licence libre et d'une licence propriétaire. Ainsi, Sun Microsystems, pour son langage Java souhaite, pour certaines raisons stratégiques, accorder une licence gratuite aux utilisateurs de Java, si les produits qui en sont dérivés sont à usage personnel ou distribués gratuitement. En revanche, elle impose une clause particulière imposant le versement de royalties au cas où le licencié désirerait tirer profit de la vente de produits dérivés de Java. Ainsi, les progrès technologiques sont encouragés par Sun Microsystems, mais à partir du moment où le licencié souhaite profiter de ces progrès technologiques, les conditions de licence changent.

Création de coopérations sources d'éventuelles économies de R&D.

étant donné le caractère coûteux, rapide et incertain de l'innovation, certaines entreprises peuvent avoir intérêt à développer une stratégie de coopération. Dans le cas d'un processus d'innovation incrémental et cumulatif, la production de l'innovation est source d'effets de feed-back. La recherche de position de leader mène alors soit à des stratégies d'intégration, soit à des accords de coopération dans le cadre de relations verticales. Dans ce dernier cas, c'est un contrat de co-production qui se met en place, l'entreprise fournissant la technologie au licencié : celui-ci se positionne alors comme un sous-traitant chargé de produire des actifs complémentaires. Le concédant profite de retours d'expérience du licencié et préserve son avantage en termes d'innovation en imposant des clauses de grant-back et la fourniture de prestations d'assistance et de conseil. Ceci permet à la R&D du concédant de profiter des effets de rétroaction de l'activité de production du licencié.

Considérant le coût de la production de logiciel, on comprend qu'une entreprise puisse décider d'encou-

rager une communauté de programmeurs à travailler sur un de ses logiciels de façon à enrichir et déboguer celui-ci. Elle va alors se servir d'une licence dont les clauses fourniront de fortes incitations à coopérer pour les programmeurs et encourageront la diffusion ; ces clauses peuvent être la libre utilisation, libre modification, libre distribution, ainsi que l'accès au code source, comme dans la GPL. Dans ce cas, l'entreprise se place dans la position d'un concédant de licence sur une technologie : le logiciel ou le langage. La «communauté» des programmeurs tient alors le rôle des licenciés, pouvant être vus comme des sous-traitants dans un contrat de co-production, pouvant produire des actifs complémentaires tels que des produits dérivés ou des versions améliorées ou déboguées. Dans ce cas, l'entreprise bénéficie d'effets de rétroaction, dans la mesure où elle profitera de ces améliorations sur son propre produit, sans engager d'efforts de R&D.

L'établissement de standard.

Dans le cas où de fortes externalités de réseau entre technologies ou utilisateurs de technologie caractérisent un marché, il peut exister des mécanismes de rendements croissants d'adoption, nous l'avons rappelé. Certaines stratégies vont donc avoir pour but de tirer parti de ces rendements croissants d'adoption afin d'acquérir une position de leader en faisant de sa technologie un standard de fait, sur un secteur où la règle celle du «gagnant prend tout». Le raisonnement est simple : plus la technologie sera diffusée, plus elle sera adoptée par un grand nombre d'acteurs, et plus elle sera attractive pour les autres acteurs qui verront là un gage de pérennité et de sécurité. Les stratégies d'ouverture, que nous avons présentées précédemment, peuvent permettre de remporter une guerre des standards.

Dans l'industrie informatique, les entreprises tentent, par divers moyens, de faire accepter leur logiciel, leur langage ou leur technologie informatique par le plus grand nombre d'utilisateurs, afin de les imposer comme standards, au détriment de ceux de leurs concurrents. Pour ce faire, elles peuvent choisir de pratiquer une politique de licence attractive et ouverte, c'est-à-dire de concéder des licences à un maximum d'utilisateurs, à des conditions qui leur feront choisir la technologie en question plutôt que celle des concurrents. Dans le cas des logiciels ou des langages, ces conditions sont traduites par des clauses de prix faible voire nul, ainsi que de disponibilité du code source et de libertés de modification. Les entreprises optant pour ce type de stratégie choisissent le plus souvent des licences soit propriétaires mais gratuites, soit libres, soit hybrides. Les exemples sont nombreux : Sun Microsystems pour Java, Troll Tech pour Qt, etc ? Comme dans le cas des technologies classiques, ce type de stratégie est souvent utilisé par les éditeurs de logiciels pour réaliser des profits joints, le plus généralement en fournissant des services, de la maintenance, des logiciels complémentaires soumis à licence propriétaire ou bien en créant un système de licence multiple en fonction du type d'activité de l'utilisateur (Troll Tech avec Qt par exemple).

8.3 Vers une méthodologie d'analyse des licences.

Il paraît évident que la stratégie d'une entreprise s'apparente rarement à l'un des quatre cas «purs» que nous venons de décrire, mais consiste souvent en un «dosage» de composantes stratégiques empruntées à ces quatre types.

Si l'on considère que chaque clause d'une licence correspond à un apport stratégique précis pour la firme, on peut voir la licence informatique comme un assemblage de ces clauses spécifiques, et donc un assemblage de composantes stratégiques empruntées aux quatre types de stratégie différents. Il apparaît donc possible de déchiffrer la stratégie globale d'une entreprise en analysant les différentes clauses présentes dans les licences de ses produits et en les couplant aux composantes stratégiques qu'elles servent.

Cette constatation conduit directement à un début de méthodologie d'analyse stratégique des licences, que nous baserons sur les quatre composantes stratégiques mises en évidence plus haut. Elle a pour principe d'associer à chaque clause d'une licence une ou plusieurs composantes stratégiques qu'elle sert. Nous allons appliquer cette méthodologie à deux exemples concrets : le cas de la librairie Qt de Troll Tech et celui du langage Java de Sun Microsystems. Pour plus de simplicité, nous symboliserons par la suite les quatre composantes stratégiques de la façon suivante : S1 pour la valorisation patrimoniale, S2 pour le contrôle des firmes concurrentes, S3 pour la création de coopérations et S4 pour l'établissement de standard.

8.3.1 Le cas de la librairie Qt de Troll Tech.

Qt est une bibliothèque mise au point par la société Troll Tech. Elle est utilisée par KDE pour son «desktop environment[5]». Le projet KDE a pour vocation de produire un desktop environment libre, placé sous GPL, de façon à offrir aux logiciels libres un environnement libre et aussi convivial que celui des environnements propriétaires. Cependant, Qt, la bibliothèque utilisée par KDE n'avait pas été placée par Troll Tech sous licence libre, ce qui constituait une entorse à «l'esprit» de la communauté du libre et provoqua de virulentes critiques de la part de celle-ci. Depuis, la licence de Qt a évolué : nous diviserons donc l'histoire de Qt en trois phases, et nous appliquerons notre méthodologie à chacune de ces trois phases.

Phase n°1.

Il convient tout d'abord de rappeler que Troll Tech applique, pour Qt, une politique de licences multiples. Elle justifie celle-ci par le fait de désirer avoir d'une part une politique libre pour son produit, mais d'autre part une politique commerciale afin de bénéficier des retombées économiques qui seraient réalisées par d'autres entreprises à partir de produits propriétaires dérivés de Qt. Ainsi, elle a créé, à cette occasion, une licence pour l'utilisation non commerciale, la «Troll Tech Free Software License» et une licence propriétaire pour toutes les entreprises qui créeraient des produits dérivés de Qt propriétaires (au code source non disponible). La version Troll Tech FSL de Qt est donc directement destinée à être «mixée» à des produits libres, alors que la version propriétaire sera «mixée», elle, à des logiciels non libres.

Si le nom même de la Troll Tech FSL pouvait indiquer qu'elle conférait un caractère libre à Qt, en réalité, sans être une licence propriétaire, elle n'était pas non plus une licence libre : si la libre utilisation était respectée, en revanche, la modification était interdite et la libre diffusion limitée aux produits dérivés dont le code source était disponible. Si l'on reprend les clauses caractéristiques de cette combinaison de licences et qu'on les associe aux composantes stratégiques qu'elles servent, on obtient le résultat suivant :

➢	Libre utilisation	➜	S4	
➢	Modification interdite	➜	S2	(va à l'encontre de S3)
➢	Disponibilité du code source	➜	S4	
➢	Distribution autorisée	➜	S4	
		+		
➢	Licence propriétaire	➜	S1 + S4	

Il apparaît clairement que l'établissement de Qt en tant que standard reste l'objectif principal de Troll Tech, vu la forte présence de clauses favorisant la composante stratégique S4 : la libre utilisation dans la mesure où elle incite à l'adoption, la disponibilité du code source, qui va également dans ce sens, puisqu'elle

[5] <FONT SIZE=3><FONT FACE="Times, serif">Un «desktop environment» sert à configurer l'aspect graphique d'un écran et est composé d'un gestionnaire de fenêtre (permettant de configurer l'encadré et la gestion des fenêtres) ainsi que d'outils comme des barres ou des menus déroulants améliorant la convivialité de l'environnement.</FONT></FONT>

garantit à l'utilisateur une certaine transparence du produit, et enfin la distribution autorisée qui encourage la diffusion. Le couplage avec une licence propriétaire va également dans le sens d'une diffusion à un maximum d'utilisateurs, puisqu'elle permet à Troll Tech de ne pas se réserver uniquement la communauté libre et de ne pas exclure de ses utilisateurs les entreprises désirant produire du non libre à partir de Qt. à l'époque, Qt et Troll Tech étant peu connus, on comprend aisément les objectifs prioritaires de diffusion et de démocratisation pour faire de cette librairie un standard. D'autre part, le contrôle des firmes concurrentes semble avoir été, lors de cette phase n°1, une autre des préoccupations principales de Troll Tech. En effet, la clause d'interdiction de modification renvoie directement à S2, étant donné qu'elle supprime toute possibilité pour quelque autre individu ou entreprise que ce soit de cloner la librairie. Si on se replace dans la situation de Troll Tech cette décision est compréhensible : étant une petite entreprise proposant un produit peu connu, elle ne pouvait courir le risque de voir une entreprise plus puissante cloner son produit et l'imposer comme standard grâce à des moyens financiers et une notoriété plus importants. Enfin, la dernière composante favorisée par Troll Tech est S1, la valorisation patrimoniale des droits de propriété intellectuelle sur Qt. La présence de la licence propriétaire destinée aux utilisateurs désirant produire des logiciels non libres dérivés de Qt lui permet en effet de bénéficier de retombées financières à travers les redevances versées. Il est important de noter l'absence de la composante stratégique S3, c'est-à-dire la création de coopérations, qui n'est représentée par aucune des clauses de la Troll Tech FSL. On peut l'expliquer par la priorité accordée notamment aux stratégies d'établissement de standard (S4) et de contrôle des concurrents (S2), nécessaires dans une phase de démarrage, parfois au détriment de la composante «coopération» (S3). La clause relative à l'interdiction de modification en est d'ailleurs la preuve, puisqu'elle permet de contrôler les concurrents, mais au prix de la privation des enrichissements susceptibles d'être effectués par les utilisateurs (S3).

Phase n°2.

KDE censé promouvoir le libre, la communauté du libre s'insurgea rapidement contre le fait que la librairie Qt soit un produit non libre. Très rapidement, cette communauté exerça des pressions sur Troll Tech pour que celle-ci remplace sa licence Troll Tech FSL par la GPL, en mettant en avant l'argument que l'interdiction de modification de la bibliothèque était un handicap sérieux pour le développement de logiciels performants. L'une de ces pressions fut notamment le lancement du projet GNOME, destiné à mettre au point un desktop environment totalement libre basé sur une librairie nommée GTK, équivalente à Qt, mais sous licence GPL. Troll Tech ne céda pas et modifia la licence protégeant Qt non pas en GPL, mais en QPL (Qt Public License), une licence toujours non libre, mais autorisant les modifications sous forme de «patches». D'autre part, elle rajouta une clause stipulant que ces modifications pouvaient être réutilisées par Troll Tech dans ce même produit ou d'autres produits, sous des licences différentes.

Voici, pour la nouvelle combinaison de licences, la représentation de chaque clause spécifique associée à la (aux) composante(s) stratégiques(s) servie(s) :

- Libre utilisation ➜ S4
- Modifications autorisées si «patches» ➜ S2 + S3 + S4
- Disponibilité du code source ➜ S4
- Distribution autorisée ➜ S4
- Modifications utilisables par Troll Tech ➜ S3

+

- Licence propriétaire ➜ S1 + S4

Avec cette nouvelle combinaison de licences, la composante S4 reste privilégiée grâce à la conservation des clauses de libre utilisation, de disponibilité du code source, de distribution autorisée et la licence propriétaire. Elle est encore renforcée par l'autorisation de modifications sous forme de «patches», dans la mesure où cette clause constitue une incitation à l'adoption de Qt par des utilisateurs qui auraient, dans la phase

précédente, été rebutés par l'impossibilité totale de modifier la librairie. La composante S2 reste également présente à travers cette clause, puisque cette solution apportée à la volonté des utilisateurs de pouvoir modifier la librairie n'entame pour autant pas le contrôle que peut exercer Troll Tech sur ses concurrents : le clonage reste toujours impossible. La composante S1 est conservée grâce à la licence propriétaire. Enfin, l'autorisation de modification est aussi l'apparition, dans la licence, de composantes S3 : la coopération d'utilisateurs enrichissant Qt ainsi que d'autres produits éventuels de Troll Tech est un aspect stratégique cette fois non négligé par la firme.

Phase n°3.

En 2001, Troll Tech décide finalement de placer Qt sous **GPL**, tout en gardant le principe de couplage avec une licence propriétaire. Voici les conséquences de ce changement de licence en termes stratégiques :

➢ Libre utilisation ➜ S4
➢ Libre modification ➜ S4 + S3
➢ Disponibilité du code source ➜ S4
➢ Distribution autorisée ➜ S4
+
➢ Licence propriétaire ➜ S1

La composante S4 continue à être largement représentée par la libre utilisation, la disponibilité du code source ainsi que la distribution autorisée. De plus, elle prend encore de l'importance de par la liberté de modification conférée par la licence GPL, au contraire de la licence précédente.

S1 reste également présente à travers les redevances versées dans le cadre de la licence propriétaire.

Mais Troll Tech renforce la composante S3 de sa stratégie, car si elle a supprimé la clause l'autorisant à disposer librement des modifications apportées à Qt, elle profite toutefois, à travers la clause de libre modification, des enrichissements effectués par les utilisateurs de Qt. On peut voir dans cette évolution stratégique la volonté de Troll Tech de voir sa bibliothèque enrichie par une communauté de programmeurs devant une concurrence potentielle de GTK de GNOME : Troll Tech peut avoir été effrayée par le desktop environment concurrent de KDE et décidé d'accéder à la demande de la communauté libre pour éviter de subir une perte de parts de marché au profit de GNOME, le desktop, même si, KDE avait conquis un grand nombre d'utilisateurs, qui ne se seraient pas tournés vers GNOME aussi facilement, que Qt ait été licencié sous GPL ou non. En effet, Troll Tech pouvait craindre que sa bibliothèque ne devienne inférieure à celle de son concurrent, constamment améliorée, elle, par une communauté de programmeurs. Inciter la communauté à améliorer Qt en la plaçant sous GPL pouvait être un bon moyen de n'être pas dépassée techniquement.

Enfin, on note la disparition de la composante S2 dans la stratégie de licence de Troll Tech qui, en adoptant la GPL, renonce au contrôle de ses concurrents en abandonnant l'interdiction de modification. En effet, autrefois une petite firme, peu connue et peu solide financièrement, Qt est aujourd'hui un quasi-standard et Troll Tech a acquis une notoriété et une maturité suffisantes pour céder à la communauté du libre et pouvoir faire passer Qt en GPL, sans craindre le risque d'une version concurrente de la part d'une grosse entreprise.

En résumé, dorénavant, pour Troll Tech, la GPL représente plus un avantage qu'un inconvénient, dans la mesure où elle incite à l'utilisation de Qt (notamment auprès d'étudiants réalisant des projets sous Qt et imposant ensuite sa version propriétaire à leurs employeurs) et dynamise par-là même les ventes de la licence commerciale, en asseyant un peu plus le standard et en enrichissant le produit du travail des programmeurs.

8.3.2 Le cas du langage Java de Sun Microsystems.

La licence SCSL a été créée par la société Sun Microsystems pour Jini, protocole Java de communication entre périphériques, puis appliquée à plusieurs autres produits, notamment Java.

Si Sun Microsystems communique sur le caractère libre de la licence SCSL, celle-ci ne s'apparente cependant pas, selon les critères de la Free Software Foundation ou OSI, à un logiciel libre, dans la mesure où les 3 critères de base ne sont pas tous strictement respectés. En effet, si le produit peut être librement utilisé et copié par tout utilisateur, c'est relativement aux conditions de distribution que des restrictions importantes sont posées :

– d'une part, la distribution par des tiers de produits dérivés de ceux protégés par la SCSL n'est pas libre : la distribution reste libre si elle est faite gratuitement, mais elle ne l'est plus du tout dès qu'un agent désire tirer un profit de cette activité. Dans ce dernier cas, Sun Microsystems impose le versement de royalties, mais s'octroie également un pouvoir de certification des produits dérivés, sans laquelle l'entreprise n'a pas le droit de distribuer son produit. Ainsi, Sun Microsystems garde un droit de regard sur tous les produits mis au point à partir de son protocole Java ;
– d'autre part, la restriction suivante est spécifiée : si les corrections d'erreurs doivent obligatoirement être placées sous licence SCSL pour être remises à la communauté, les modifications d'autres sortes (comme des améliorations) peuvent devenir propriétaires de Sun, même dans le cas où elles seraient le fruit du travail de membres de la communauté de développeurs ne travaillant pas pour Sun.

➢ Libre utilisation	➜	S4
➢ Libre modification	➜	S4 + S3
➢ Distribution gratuite libre	➜	S4
➢ Redevance	➜	S1
➢ Certification	➜	S2
➢ Modifications utilisables	➜	S3

La SCSL de Sun Microsystems favorise largement la composante S4 à travers les clauses de libre utilisation, libre modification et libre distribution si celle-ci est gratuite. Ces trois clauses traduisent ainsi la volonté de diffuser le plus largement possible Java, de le démocratiser et d'inciter à son adoption. On peut penser que cet aspect stratégique a été d'une grande importante dans la mise de Java sous cette licence. Cette licence, contrairement à une licence libre de type GPL, permet également de combiner S4 avec la composante S1 de valorisation patrimoniale. Pour cela, elle pose une discrimination entre utilisateurs et différencie les utilisateurs qui vont redistribuer gratuitement les produits dérivés, et ceux qui vont tirer un bénéfice financier de la vente de produits dérivés de Java. Pour ces derniers, la redistribution n'est pas gratuite, mais soumise à redevance à Java, qui impose également une clause de «grant-back», c'est-à-dire le versement de royalties sur les bénéfices réalisés sur ces ventes.

Par la clause de certification, c'est S2, le contrôle des firmes concurrentes, que Sun Microsystems met en place. En exigeant que tout produit dérivé destiné à être vendu lui soit présenté afin d'être approuvé et d'obtenir la certification Java ou pas, Sun garde le contrôle de l'évolution de ses produits et de l'utilisation qui peut en être faite par ses concurrents, éliminant ainsi les projets qui iraient à l'encontre de la stratégie de la firme. Enfin, cette licence ne néglige pas non plus la composante S3, dans la mesure où la clause de libre modification instaure une logique de coopération des utilisateurs de Java, permettant de l'enrichir constamment. Cette coopération est d'autant plus bénéfique pour Sun que la clause l'autorisant à utiliser ces modifications dans quelque produit que ce soit et sous quelque licence que ce soit, permet à Java d'enrichir, par le travail de la communauté de programmeurs non seulement Java mais tous ses produits, y compris ceux soumis à des licences propriétaires. Contrairement à l'exemple de Qt, pour lequel la licence avait évolué

au fur et à mesure de la maturité de Troll Tech, Sun Microsystems, malgré la pression de ses concurrents et de la communauté du libre, inquiets de voir un standard contrôlé par une seule entreprise, n'a, à ce jour, pas accepté de placer Java sous une licence libre. En effet, le système adopté n'étant pas celui de la double licence, placer le produit sous GPL par exemple n'autoriserait pas Sun à conserver les composantes S1 et S2, ce qui la priverait d'une partie des bénéfices réalisés sur les ventes de produits dérivés.

L'étude de ces deux exemples montre bien quel peut être l'intérêt, pour les entreprises, d'utiliser une licence de type hybride : celui de ménager, dans une même licence, ou dans un système de double licence, des intérêts qui seraient incompatibles si l'on utilisait une licence libre pure ou une licence propriétaire pure. Les licences hybrides naissent donc de la volonté de profiter de tous les avantages procurés par une licence propriétaire, liés au contrôle de la technologie et à la valorisation patrimoniale, ainsi que de tous les avantages d'une licence libre, liés à la diffusion, à la standardisation et à l'enrichissement du produit. Les licences hybrides permettent également d'échapper aux inconvénients de chacun de ces types de licences : l'absence de retombées financières directes et l'absence de contrôle pour les licences libres, et la difficulté de diffusion et d'adoption ainsi que l'absence d'enrichissement du produit pour les logiciels propriétaires. Si l'on se réfère aux deux stratégies mises en avant par Shapiro et Varian [2000], on s'aperçoit que les licences hybrides permettent à l'entreprise de concilier stratégie de contrôle et stratégie d'ouverture.

8.3.3 Stratégie et choix de licence : une vision dynamique à privilégier.

Après avoir appliqué cette méthodologie à diverses licences, on peut penser que son intérêt consisterait à créer deux types d'outils. Le premier serait un «catalogue» associant aux différents types de clauses recensés la ou les composantes stratégiques servies. Il aurait pour but de «traduire» toute licence d'un langage juridique dans un «langage stratégique», ce qui permettrait à tout licencié de cerner très rapidement les buts de l'entreprise avant d'adopter son logiciel. Le deuxième prendrait également la forme d'un «catalogue», mais qui recenserait cette fois les différents types de clauses susceptibles de servir chacune des composantes stratégiques identifiées. Cet outil serait utilisé par toute entité souhaitant trouver la licence appropriée à son produit et à ses buts stratégiques.

Cependant, cette méthodologie et ces outils procureraient une vision statique d'une licence, incohérente dans la mesure où le propre d'une stratégie est d'évoluer au cours du temps, en fonction de nombreux facteurs (l'évolution de ses parts de marché, de sa notoriété, de la situation dans laquelle se place son produit, phase de démarrage ou position de quasi-standard, du nombre et du pouvoir des concurrents, ou enfin des actions et pression d'acteurs institutionnels tels que les associations de promotion du Libre, par exemple). Cette vision trop statique peut même devenir dangereuse pour une entreprise, celle-ci se devant de toujours anticiper les évolutions de son marché, de l'usage de son logiciel, donc de sa licence pour ne pas se retrouver bloquée dans une stratégie figée.

Ainsi, il semble évident que la stratégie d'une firme ne pourra être la même lorsque le produit sera en phase de démarrage ou en phase de maturité. Lors de la phase de démarrage, le but premier de l'entreprise sera de faire adopter le produit par un maximum d'utilisateurs, de façon à l'imposer en tant que standard. Pour ce faire, l'éditeur pourrait choisir une licence aux clauses peu restrictives et imposant une faible redevance. En revanche, lorsque le produit atteint une phase de maturité où il est établi comme standard, la stratégie de l'éditeur ne visera plus une standardisation déjà acquise, mais plutôt la rentabilisation, le contrôle des firmes concurrentes ou l'enrichissement du logiciel par une communauté de programmeur. Il est alors probable qu'en fonction de l'objectif choisi, les clauses de la licence précédente ne seront plus appropriées.

De même, l'environnement concurrentiel de l'entreprise peut se modifier. Une entreprise en situation de quasi-monopole sur son créneau aura tendance à adopter une stratégie de valorisation et en conséquence une

licence plutôt restrictive et à redevance élevée. à partir du moment où des concurrents font leur entrée sur son marché, et qu'elle voit ses positions menacées, elle aura tout intérêt à favoriser une stratégie de contrôle des concurrents ou de standardisation : là encore, la licence adoptée précédemment risque alors de ne plus être appropriée. Enfin, comme nous l'avons vu dans le cas de la librairie Qt de Troll Tech, une stratégie peut être amenée à évoluer sous la pression d'acteurs institutionnels tels que les associations de promotion du Libre comme la FSF. Certaines entreprises peuvent alors être poussées à revoir leur licence et par-là même leur stratégie.

Or, de même qu'il existe des dépendances de sentier technologiques, c'est-à-dire qu'un choix initial d'une technologie a des conséquences sur les possibilités de choix futurs en rendant impossible ceux de certaines technologies, de même ces dépendances de sentier sont susceptibles d'être présentes au niveau des licences : en fonction des clauses choisies, une firme peut s'engager dans un sentier et se priver de la possibilité de modifier ou rajouter certaines clauses afin de faire évoluer cette licence.

Les sources de cette dépendance de sentier de licence sont les facteurs défavorisant le changement de licence. On en compte quatre.

La première est d'ordre juridique, dans la mesure où les clauses mêmes d'une licence peuvent interdire de placer le produit sous quelque autre licence que ce soit sans l'accord de tous les coauteurs, qui peuvent être très nombreux : c'est le cas de la GPL, qui stipule que tout produit GPL ou dérivé d'un produit GPL ne peut être placé sous une autre licence que la GPL.

La deuxième source est relative à la popularité du produit et à son éventuelle position de standard : comme une licence peut s'avérer incitative à l'adoption d'un produit, sa modification en une version plus stricte est susceptible, au contraire, de conduire les utilisateurs à se tourner vers un autre produit.

La troisième concerne les institutions et lobbies, tels que les associations de promotion du libre de type Free Software Foundation. Celles-ci ont en effet tendance à encourager les entreprises éditrices de produits standards à leur appliquer des licences libres : tout retour vers une licence de type hybride donnerait lieu à de fortes pressions de leur part.

Enfin, la quatrième est relative à la communication : changer de licence implique d'importants efforts de communication visant à expliquer aux utilisateurs les différences avec l'ancienne version, mais aussi à les rassurer quant aux conséquences qu'un tel changement. Ces dépenses de communication sont d'autant plus indispensables que les utilisateurs sont frileux et suspicieux par rapport aux motivations de l'entreprise éditrice, qu'ils peuvent soupçonner par exemple de réduire le caractère libre d'une licence.

Il apparaît cependant que les désagréments de cette dépendance de sentier se font sentir moins fortement dans le cas où l'entreprise désire faire évoluer sa licence vers une version plus libre, que lorsqu'elle tente de la modifier dans le sens inverse, c'est-à-dire de limiter les libertés octroyées jusque là. Ainsi, dans le cas de Troll Tech, qui a choisi de faire évoluer sa licence d'une version hybride vers une version libre, et enfin vers une version GPL, l'effet de la dépendance de sentier a été plutôt peu ressenti. En revanche, on peut imaginer le cas d'une société ayant choisi comme licence, pour un de ses produits, la GPL ou la LGPL, des licences libres pures : la dépendance de sentier aurait des effets très néfastes si elle décidait de revenir à une licence hybride, dans la mesure où les 4 sources de dépendance de sentier citées plus haut rendraient très difficile et dangereux ce type de changement de licence.

Conclusion.

En nous appuyant sur des cas concrets, nous avons tenté de montrer, tout au long de cet article, que les licences sont, pour les éditeurs de logiciels, des outils pour mener à bien une stratégie définie au préalable en fonction de leur position sur le marché. La licence apparaît alors comme un assemblage de clauses servant des composantes stratégiques précises telles que la valorisation patrimoniale, le contrôle des firmes concurrentes, la création d coopérations sources d'éventuelles économies de R&D et la standardisation. Ce constat étant fait, il est alors possible de déchiffrer chaque licence de logiciel en mettant à jour les orientations stratégiques de l'éditeur en fonction des clauses choisies. En allant plus loin, il serait tentant de déduire de ces observations une méthodologie qui permettrait d'associer automatiquement chaque composante de la stratégie de la firme à une clause spécifique de la licence. Ce serait une démarche un peu hâtive, voire même inefficace sinon dangereuse à long terme. En effet, s'il est vrai que certaines licences sont à privilégier en fonction de la stratégie choisie par l'éditeur, il ne faut surtout pas perdre de vue que stratégie et licence s'opposent sur un caractère précis : l'évolutivité. Si le propre de la licence est son caractère défini et statique, il n'en va pas de même pour la stratégie : celle-ci est très souvent amenée à évoluer et se modifier au cours du temps en fonction notamment du succès du produit, de la structure du marché ou de la concurrence. On comprend alors tout le risque du choix d'une licence statique destinée à servir une stratégie vouée à évoluer : cela introduit une certaine irréversibilité et ferme des opportunités stratégiques. Dès lors, le choix de la licence ne peut plus se limiter, pour un éditeur, à traduire sa stratégie actuelle en clauses, mais demande un travail d'anticipation bien plus complexe, prenant en compte l'ensemble du cycle de vie du produit.

Troisième partie

Étude du contexte juridique de l’économie du logiciel.

L’usage des licences de logiciels libres.

YVES ROUGY,
Alcôve, Yves@rougy.net.

Introduction.

Les licences libres sont fortement dominées par la présence de la licence GPL. Cette dernière représente la grande majorité des licences libres utilisées actuellement.

Néanmoins, derrière la suprématie de la GPL, d'autres licences coexistent. Historiquement, la licence GPL a vu son utilisation exploser avec la démocratisation du système d'exploitation Linux. Mais des licences plus anciennes, comme la licence BSD ou ses dérivées restent quand même très présentes.

Lorsqu'une entreprise développe un logiciel libre, il est fréquent de voir apparaître une licence attachée à son projet, ou à elle-même. On peut d'ailleurs remarquer que ces licences comportent souvent le nom de l'entreprise dans leur propre nom.

D'une manière plus générale, on remarque que l'on peut d'ailleurs presque systématiquement associer une licence spécifique à un projet particulier, comme Perl avec la licence Artistic ou BSD avec le système Unix de l'université de Berkeley, donc également hors des entreprises.

Si Internet permet d'avoir facilement et rapidement accès à ces licences, en revanche, trouver une ancienne version, ou faire l'historique d'une licence particulière est particulièrement difficile. En effet, l'aspect immédiat d'Internet fait disparaître toute l'évolution d'une page contenant une licence par exemple. On ne peut alors accéder aux versions anciennes du document que si l'auteur met ses anciennes versions sur le site. Ainsi les versions du document sont perdues si l'auteur ne les a pas toutes archivées. Certaines informations qui auraient pu être présentées dans cette étude ne le sont pas par manque d'information historique, comme les dates de création des licences !

9.1 La liste des licences.

Les licences présentées ici ont été trouvées grâce aux moteurs de recherches, et notamment la partie annuaire de Google (directory.google.com). De plus, les informations trouvées sur le site OSI (http://OpenSource.org) ont été d'une précieuse aide.

En ce qui concerne les statistiques sur les projets, (thématiques, répartitions), seules les licences les plus représentatives ont été analysées. Ces statistiques thématiques ou de répartitions ont été faites en utilisant les données accessibles par le site http://SourceForge.net. Ce site permet à un programmeur d'héberger son projet, en spécifiant, entre autres informations, la licence qu'il utilise. Le grand nombre de projets hébergés (de l'ordre de 45000) permet d'avoir un échantillon suffisamment représentatif pour s'intéresser aux licences utilisées par ces projets. De plus, SourceForge répartit ses projets dans différents thèmes. Pour de raisons de clarté, les thèmes de SourceForge ont été regroupés dans cette étude en thèmes plus généraux. Le nombre de projets présents dans la nouvelle classification est de plus de 35000.

Si toutes les licences ne sont pas répertoriées sur le site de SourceForge, on y trouve les plus représentatives.

9.2 Provenance géographique.

La provenance géographique des licences a été obtenue en recherchant le lieu du créateur de la licence. Les informations sont en général disponibles sur le même site Web que la licence.

Pour ces statistiques, 44 licences ont été étudiées[1]. La différence avec le nombre total de licences provient de la difficulté de trouver la vraie origine des licences. Très souvent, seul le texte de la licence et éventuel-

[1] La somme des provenances géographiques est supérieure à 44 car plusieurs licences proviennent de collaborations internationales.

lement sa version sont disponibles. Parfois, ce sont les informations relatives au projet ayant entraînées la création de la licence qui permettent de déduire l'identité des équipes rédactrices.

La grande majorité des licences provient des États-Unis, avec plus de 75 % des licences. Le reste vient principalement d'Europe du nord (Près de 30 %). Il est logique de constater que les pays les plus industrialisés sont les plus producteurs de licences libres. Sur les 44 licences présentes, une seule vient d'une collaboration impliquant, outre des équipes française et américaine, une équipe asiatique (la licence du W3C).

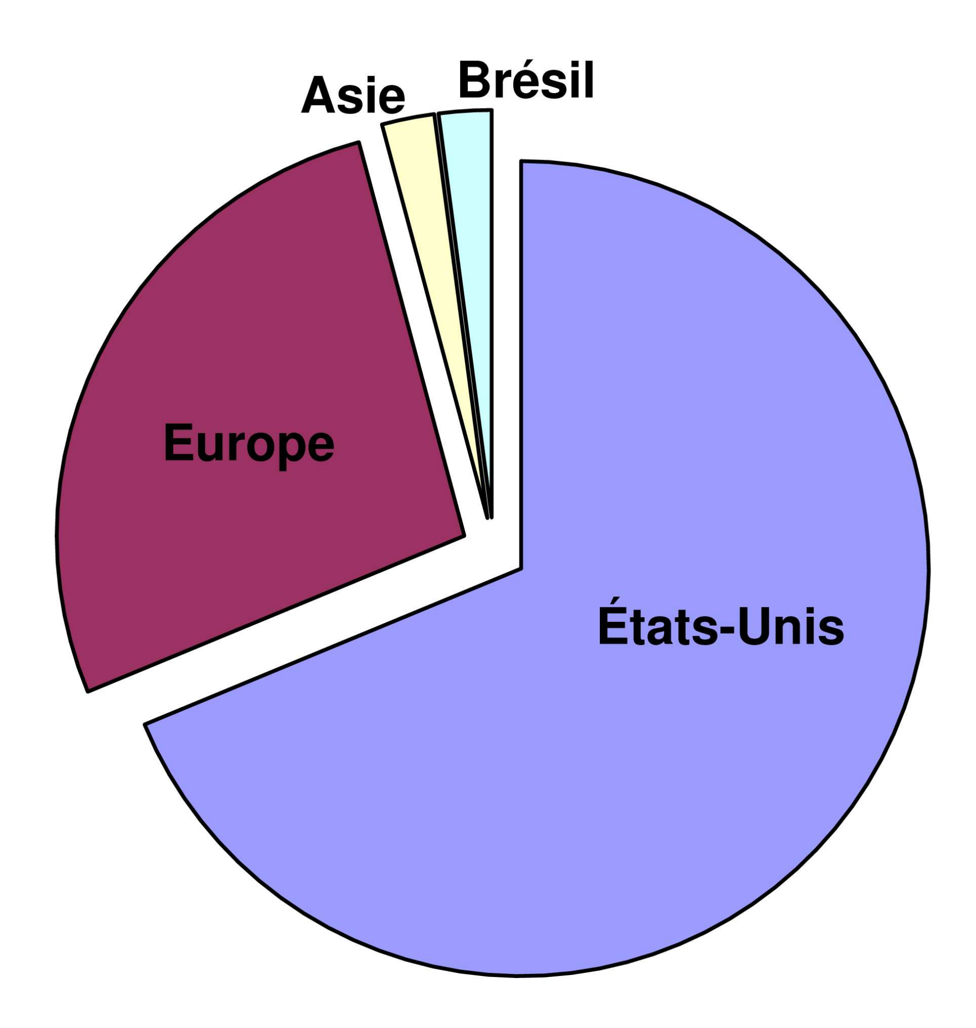


Figure 9.1 — Répartition géographique des licences.

9.3 Les auteurs des licences libres.

On a cherché à classer les auteurs des licences étudiées dans des catégories définies de la façon suivantes :

- Entreprise : c'est le cas des licences créées par des entreprises pour un projet particulier ou en suivant une politique d'ouverture. Parfois, la création de la licence fait partie des fondations mêmes de l'entreprise concernée.
- Université : les universités, et en général leurs laboratoires de recherche, qui sont très prolixes en termes de projets novateurs. Ceci a eu pour conséquence la création d'un bon nombre de licences pour diffuser ces projets.

– Association : On trouve dans cette catégorie les individus, les fondations, ou les associations créant des licences libres, comme la FSF, LaTeX, ou des associations d'entreprises.

On trouve parfois des partenariats entre universités et entreprises ou entre entreprises et associations. C'est souvent le cas dans le cadre de projets particuliers, ces derniers contenant une licence originale.

Une fois encore, 44 licences ont été étudiées pour obtenir ces chiffres. Quelquefois, il n'y a pas d'informations suffisantes pour déduire l'origine des auteurs d'une licence, certaines licences ne sont donc pas comptabilisées dans ces chiffres.

Les licences libres proviennent pour leur majorité des entreprises (22 licences, soit 50 %). Ensuite, on trouve les universités avec 15 licences (34 %) et pour finir les associations avec 10 licences (22 %). Une fois de plus, le total est supérieur à 44, ou 100 %, car les licences utilisées à l'occasion de partenariats sont comptabilisées deux fois.

L'explication de cette répartition est assez simple à comprendre. En effet, les associations ont tendance à créer des licences assez génériques et très utilisées (comme la GPL par exemple), alors que les entreprises ont tendance à créer une licence par type de projets. On trouvera souvent une ou plusieurs licences associées à une entreprise. A contrario, chaque université crée sa propre licence, et tous les projets de l'université utiliseront la même licence.

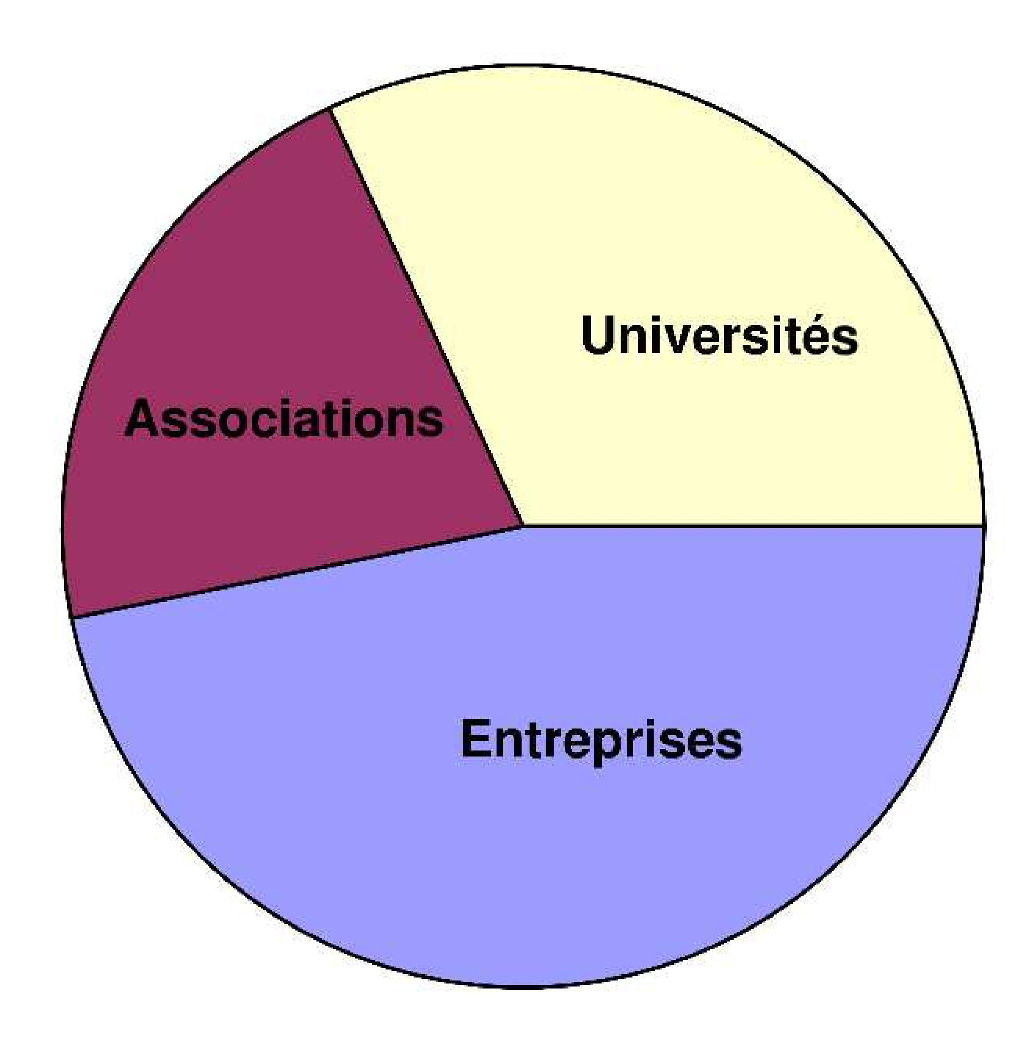


***Figure 9.2* —** Répartition des origines licences.

9.4 Répartition des licences.

La licence GPL est de loin la plus représentée, puisque sur plus de 45000 projets étudiés on trouve près de 35000 projets utilisant la GPL, soit plus de 70 %. La licence LGPL suit avec plus de 10 % des projets, soit près de 5000 projets.

La licence suivante est la licence de type BSD, qui représente environ 10 % des projets. Il est à noter que cette licence est sous évaluée, car de nombreux projets utilisent une licence de type BSD en la renommant.

On trouve ensuite la licence Artistic qui est la licence du programme Perl (ce dernier étant sous double licence Artistic/GPL), avec une représentation de moins de 3 %, suivit par la licence MIT, avec un peu plus de 1,5 %.

Comme on le voit, la répartition est assez inégale, les licences de la FSF (GPL et LGPL) représentant plus de 80 % des projets étudiés.

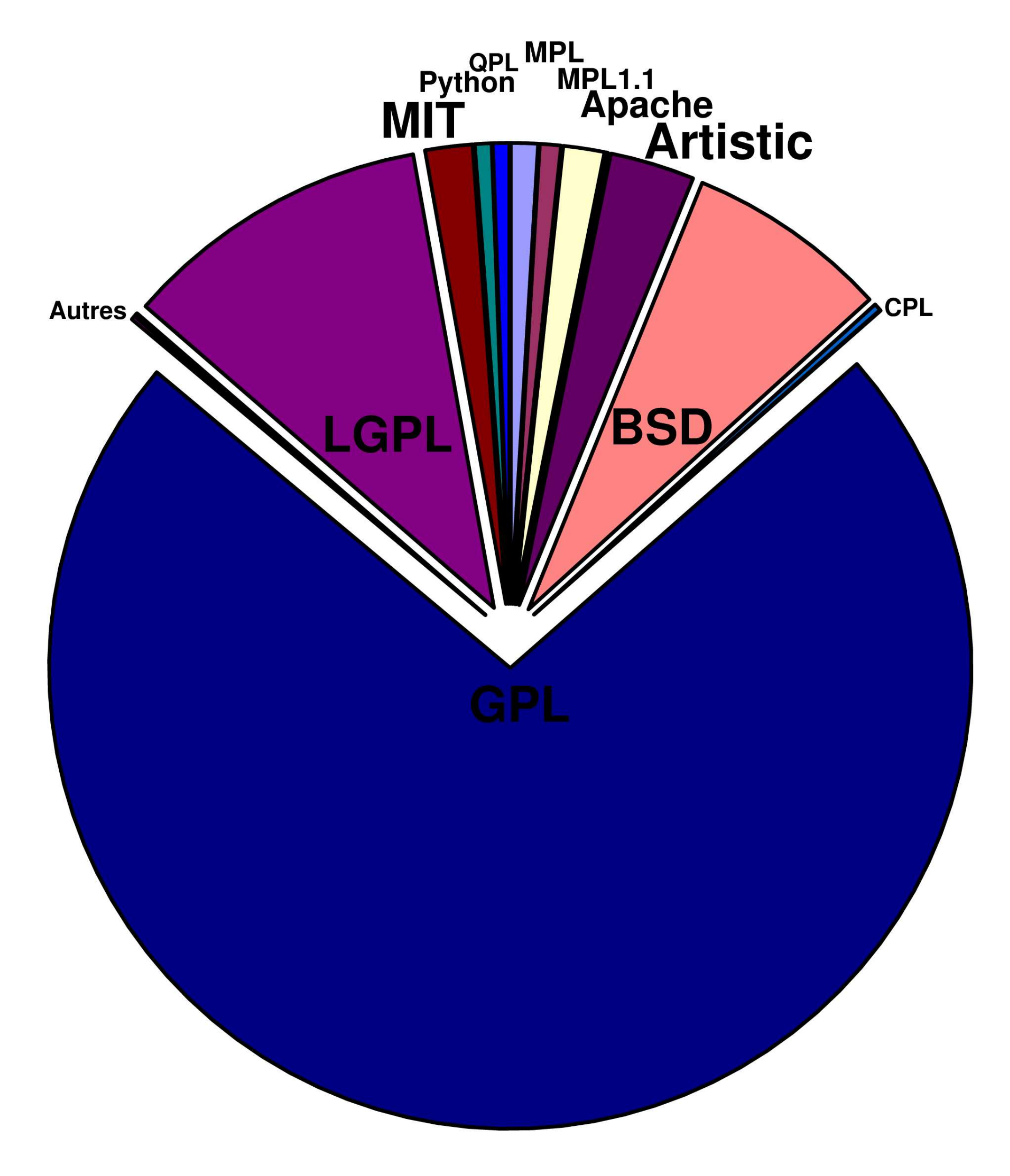


Figure 9.3 — Répartition des projets par licences.

9.5 Répartition thématique des licences.

9.5.1 Introduction.

Pour ces statistiques, on a voulu voir dans quel(s) domaine(s) les licences libres étaient les plus utilisées, mais également, pour chaque licence, quelles étaient ses thématiques dominantes.

Ces informations ne se trouvent pas sur les sites donnant le texte de la licence, mais plutôt en comptant le nombre de projets par licence, et la thématique de chaque projet. La source ayant permis d'avoir ces informations est le site communautaire SourceForge (http://sf.net).

9.5.2 Répartition générale.

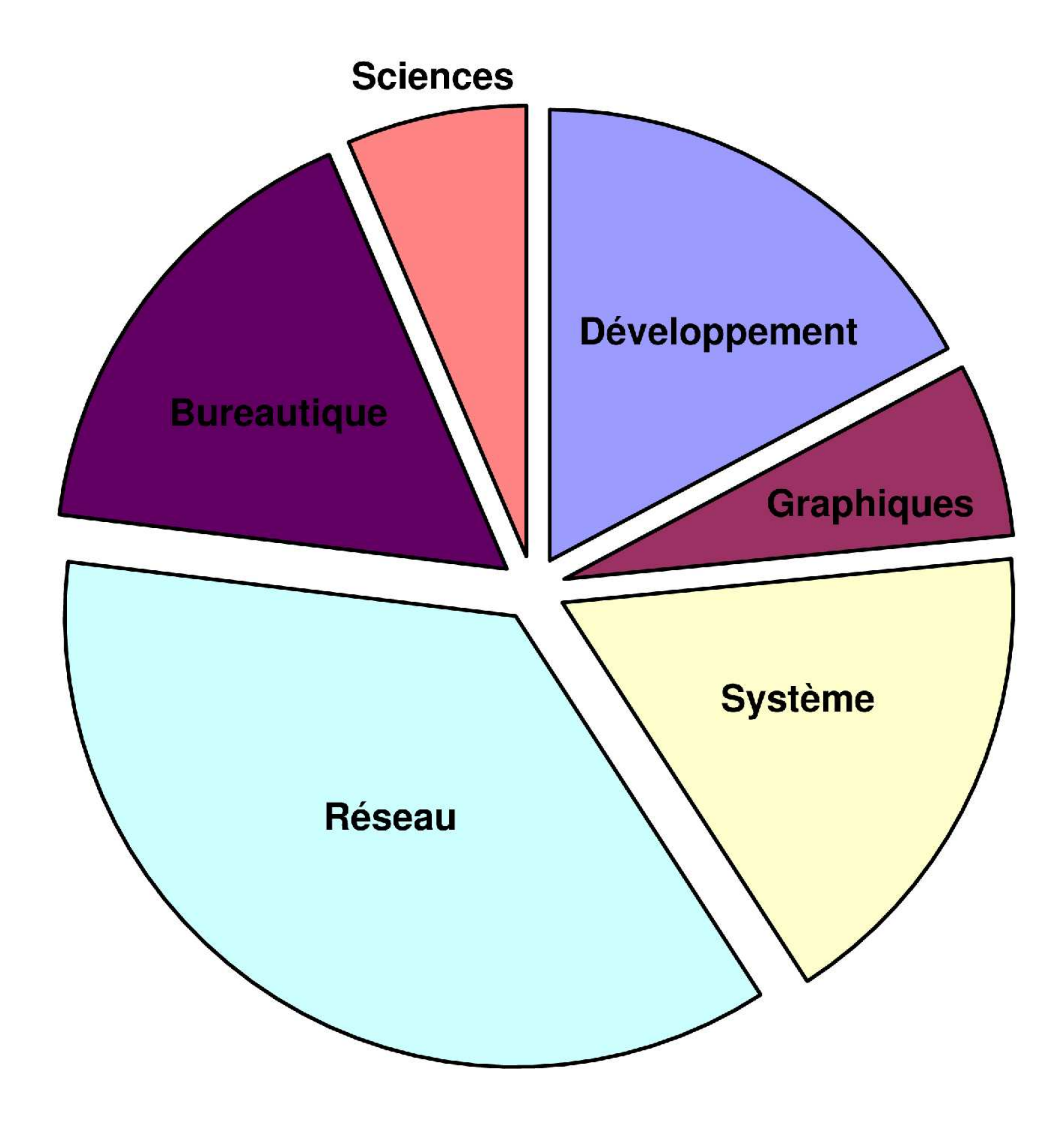


Figure 9.4 — Répartition générale par thèmes

Développement	17,2 %
Graphiques	6,2 %
Système	17,4 %
Réseau	36,0 %
Bureautique	16,6 %
Sciences	6,5 %

Il est intéressant de constater que plus d'un tiers des projets libres étudiés concernent les applications réseau. Les licences libres ont vu leur popularité grandir grâce à des médias comme Internet, et également par Internet et les applications réseau.

L'importante représentation des projets liés au développement ou au système est assez significative de l'utilisation habituelle de systèmes libres comme GNU/Linux. La présence importante de projets liés à la bureautique est plutôt récente. En effet, on a longtemps reproché l'absence de projets de bureautiques libres.

La forte présence de projets liés au réseau et au système montre également la bonne implantation des projets libres dans les applications d'infrastructure.

9.5.3 La GPL.

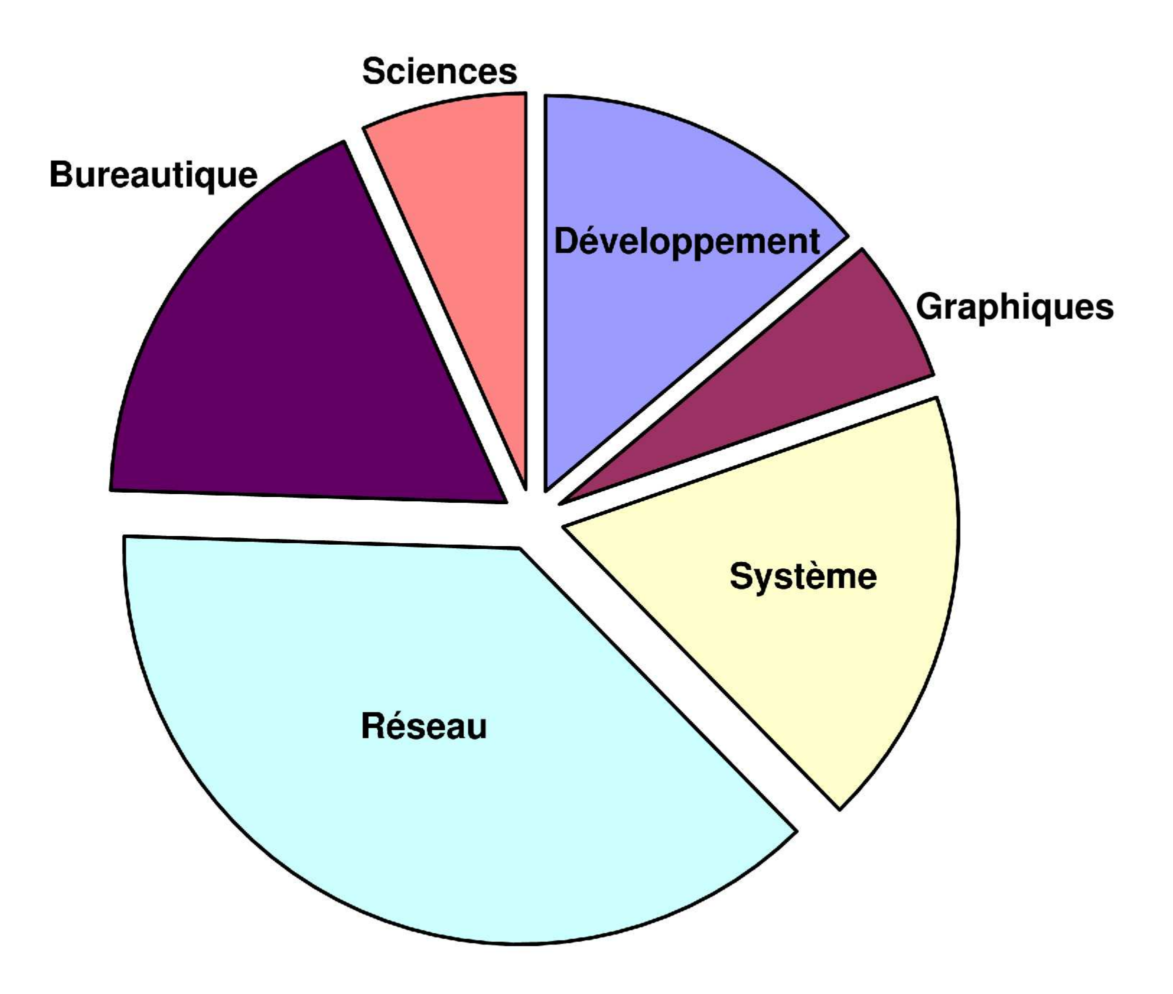


Figure 9.5 — Répartition de la GPL par thèmes.

Développement	13,8 %
Graphiques	5,8 %
Système	18,0 %
Réseau	37,82 %
Bureautique	17,8 %
Sciences	6,7 %

On retrouve une répartition assez similaire à la répartition générale : la licence GPL représente en effet les trois quarts des projets étudiés.

9.5.4 La LGPL.

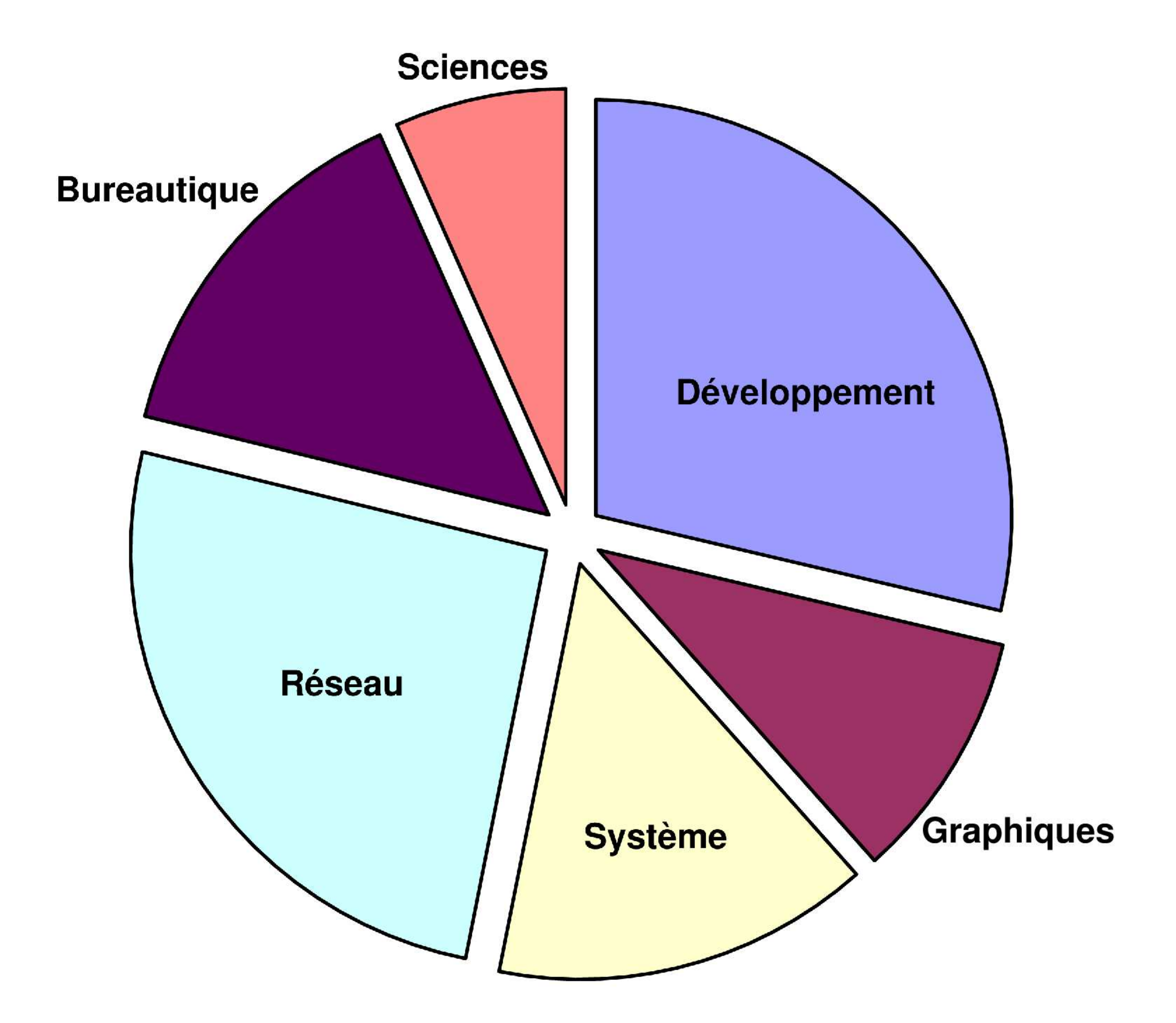


Figure 9.6 — Répartition de la LGPL par thèmes.

Développement	28,7 %
Graphiques	9,8 %
Système	14,7 %
Réseau	25,7 %
Bureautique	14,5 %
Sciences	6,7 %

On a ici une forte implication dans les activités de développement. En effet, cette licence a été écrite pour des bibliothèques de programmation, afin de permettre leur utilisation avec des logiciels dont la licence n'est pas compatible avec la GPL. Il semble que ce soit également le cas pour les projets sur les réseaux.

9.5.5 La licence BSD.

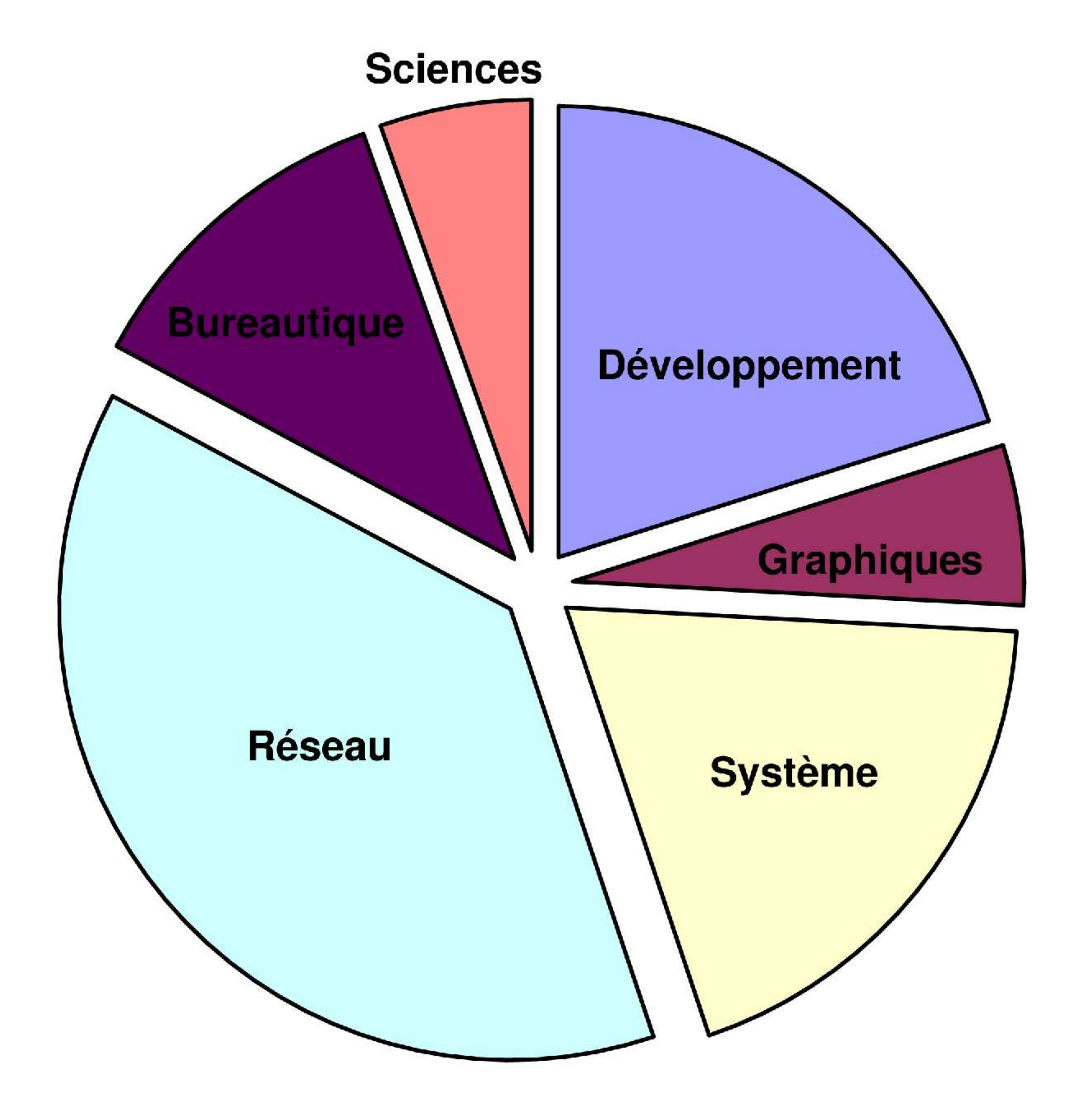


Figure 9.7 — Répartition de la licence BSD par thèmes.

Développement	20,1 %
Graphiques	5,7 %
Système	19,0 %
Réseau	38,0 %
Bureautique	11,7 %
Sciences	5,4 %

La licence BSD a une forte représentation dans les projets liés au réseau. Le fait que la couche IP des systèmes BSD ait été beaucoup réutilisée, comme le permettait sa licence, n'y est peut-être pas étranger. Viennent

ensuite les thèmes du développement et du système. L'implication de ces projets dans les infrastructures est particulièrement visible.

9.5.6 La licence Artistic.

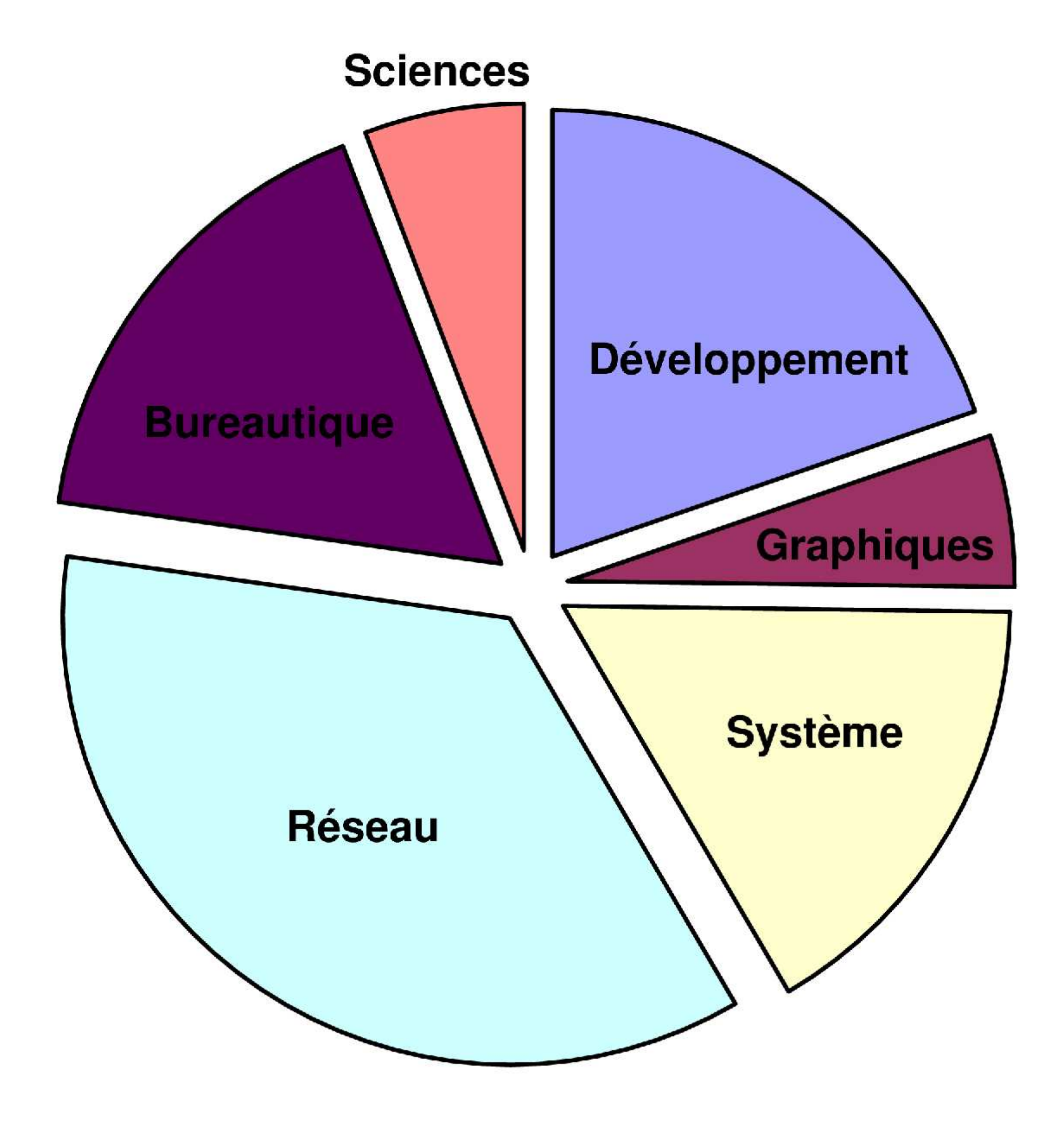


Figure 9.8 — Répartition de la licence Artistic par thèmes.

Développement	19,7 %
Graphiques	5,5 %
Système	16,4 %
Réseau	35,6 %
Bureautique	17,0 %
Sciences	5,7 %

La licence Artistic est la licence du logiciel Perl qui est très utilisée dans les applications Web et système. De plus, la modularité de Perl a permis à de nombreux projets étendant ses capacités de naître, tout en utilisant

la même licence, ce qui explique sans doute cette répartition. Cette licence est donc très présente dans les applications réseau et système.

9.5.7 La licence MIT.

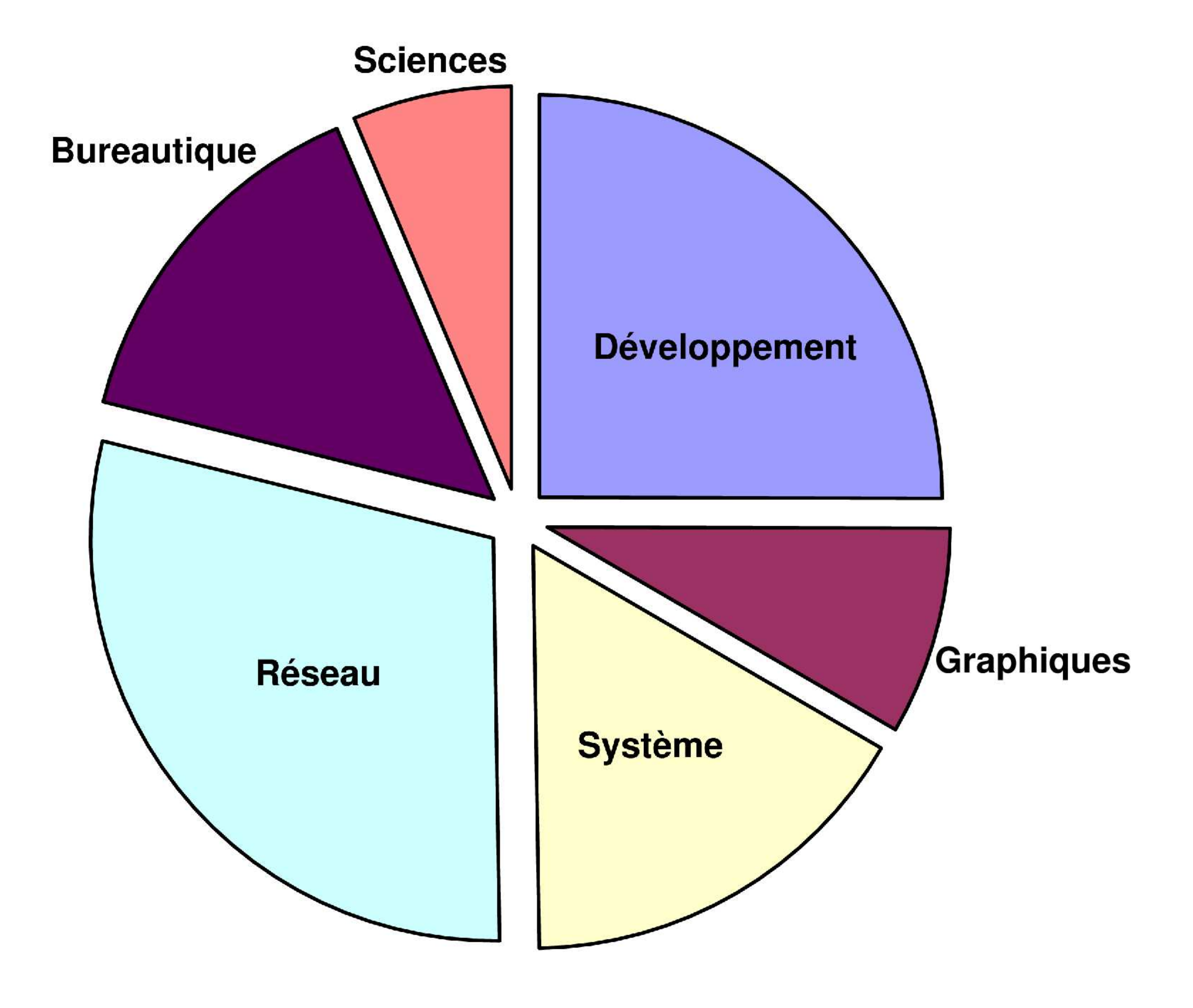


Figure 9.9 — Répartition de la licence MIT par thèmes.

Développement	25,0 %
Graphiques	8,3 %
Système	16,4 %
Réseau	29,1 %
Bureautique	14,7 %
Sciences	6,4 %

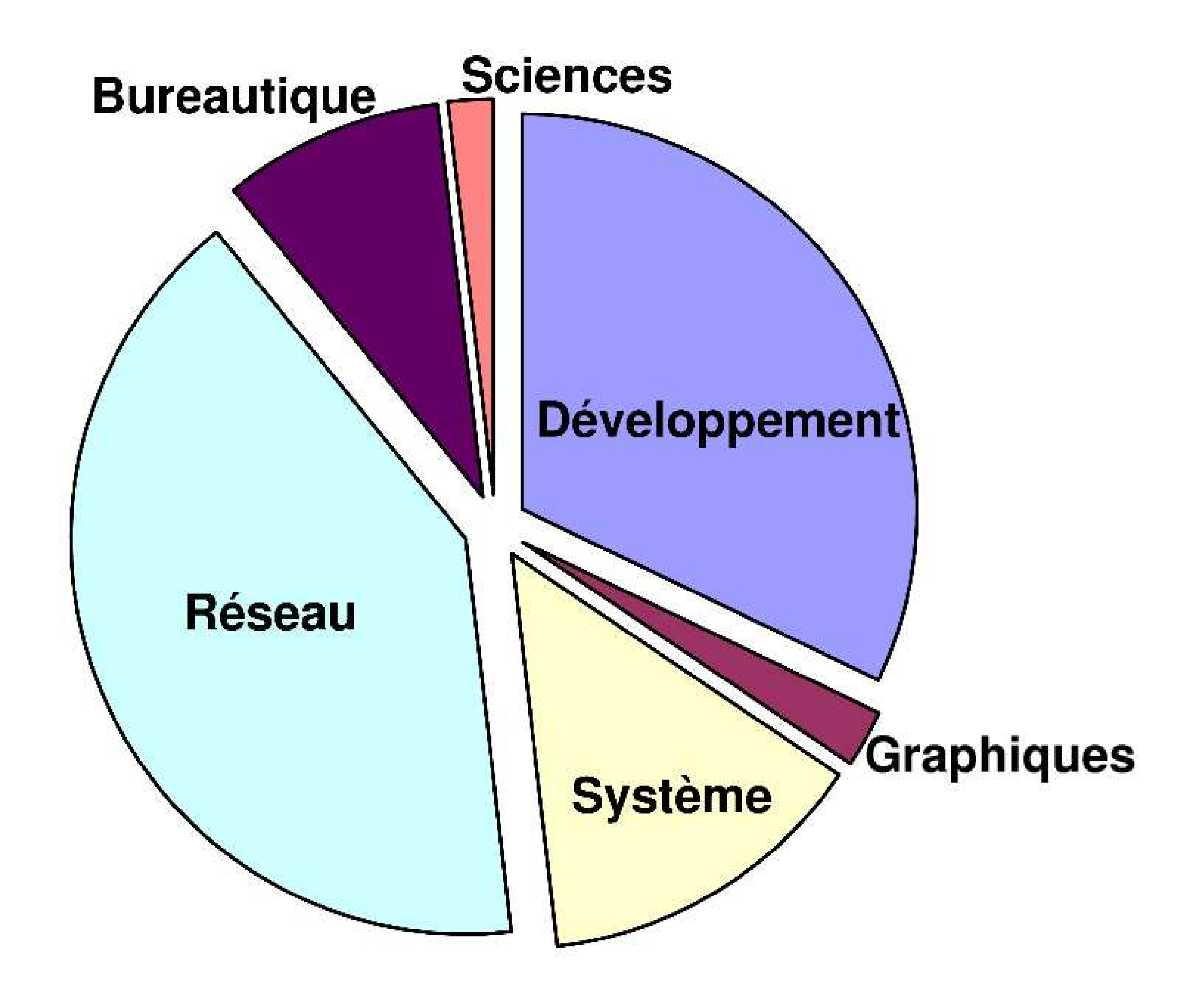


Figure 9.10 — Répartition de la licence Apache par thèmes.

9.5.8 La licence Apache.

Développement	32,1 %
Graphiques	2,3 %
Système	13,7 %
Réseau	41,0 %
Bureautique	9,0 %
Sciences	1,8 %

Cette licence concerne au départ le logiciel Apache, qui est le serveur Web le plus répandu au monde. C'est donc logiquement que l'on constate que plus de 40 % des projets l'utilisant concerne les activités réseau.

9.5.9 La licence MPL.

Développement	19,5 %
Graphiques	2,4 %
Système	21,7 %
Réseau	36,4 %
Bureautique	15,0 %
Sciences	5,1 %

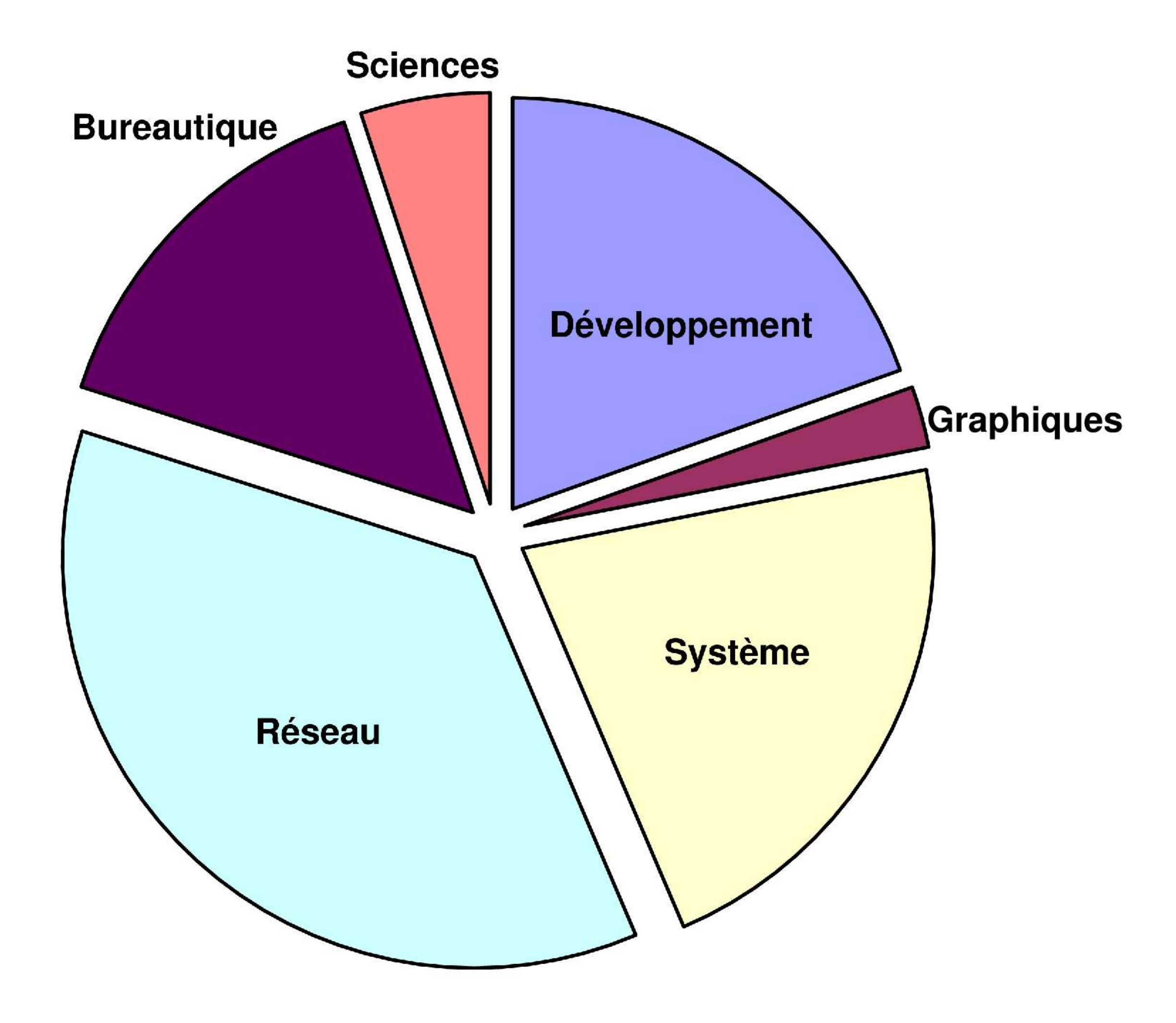


Figure 9.11 **—** Répartition de la licence MPL par thèmes.

Cette licence est la première version de la licence sous laquelle le navigateur Mozilla a été publié. Les limitations de cette licence sont assez vite apparues, et une version 1.1 créée (cf. infra). Cependant, beaucoup de projets continuent d'être publiés sous cette licence et pas sous la version 1.1.

9.5.10 La licence MPL 1.1.

Développement	28,5 %
Graphiques	5,6 %
Système	10,2 %
Réseau	35,9 %
Bureautique	14,4 %
Sciences	5,3 %

9.5.11 Les autres licences.

Chacune des autres licences étudiées représentent moins de 1 %. Compte tenu de leur faible représentativité, elles font l'objet ici, d'une étude groupée.

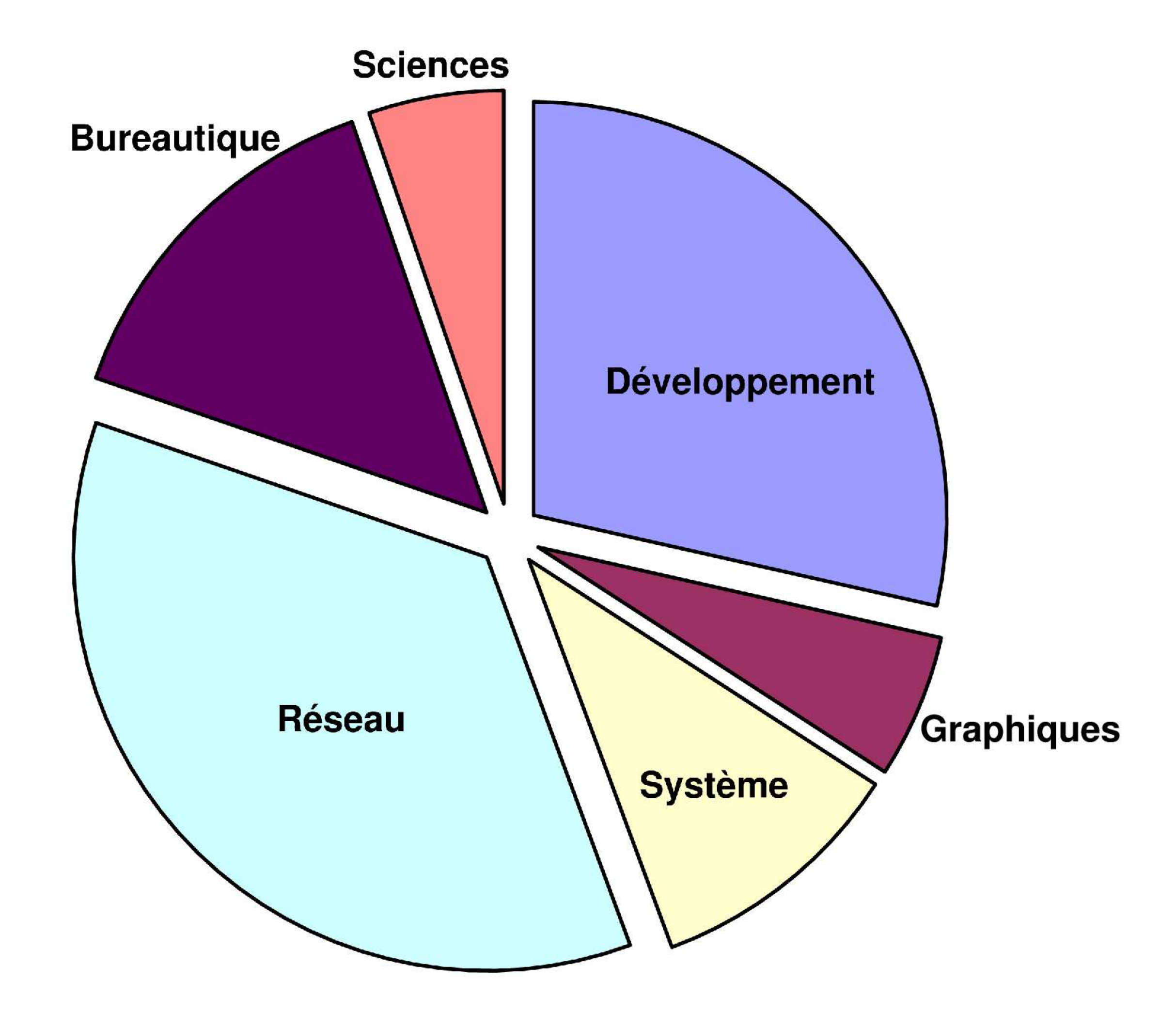


Figure 9.12 **—** Répartition de la licence MPL 1.1 par thèmes.

- APSL (Apple) : Les grandes thématiques rencontrées concernent le réseau, le développement et le système.
- CPL : Cette licence, très utilisée par IBM, concerne principalement le développement.
- IPL : Cette licence a permis la naissance de la CPL. Elle concerne le réseau, le développement et le système.
- Eiffel : Cette licence qui est associée au langage éponyme concerne principalement les activités de développement.
- Intel : Concerne principalement les activités réseau.
- Jabber : Cette licence a été introduite avec un projet très lié au réseau. C'est donc logiquement que les activités qui y sont liées concernent principalement le réseau.
- Python : La licence du langage éponyme concerne le développement, mais également, et à part égale, les activités réseau.
- QPL : Cette licence, liée aux bibliothèques d'affichage "Qt" concerne le développement, mais également le réseau, à parts égales. En raison de l'augmentation de machines de type Unix pour la bureautique, et la maturité de l'environnement KDE (qui utilise Qt), on trouve également un nombre représentatif de projets concernant la bureautique.

Les licences libres. Analyse juridique.

Mélanie Clément-Fontaine,
Alcôve, Université de Montpellier 1/ERCIM, melanie@amberlab.net.

10.1 Définitions générales.

10.1.1 La licence.

La licence est une convention par laquelle une personne (appelée le donneur de licence) autorise une autre personne (appelée le licencié) à utiliser son logiciel sans en transférer les droits qui y sont attachés.

10.1.2 Logiciel.

La directive communautaire de 1991[1] parle de programme ainsi que les dispositions de l'OMPI qui le définit comme «un ensemble d'instructions pouvant, une fois transposé sur un support déchiffrable par la machine, faire indiquer, faire accomplir ou faire obtenir une fonction, une tâche ou un résultat particulier par une machine capable de faire du traitement de l'information» (Dispositions types de l'OMPI relatives à la protection du logiciel). Toutefois, c'est le terme logiciel qui a été retenu en France et qui, selon les mêmes dispositions de l'OMPI, couvre en outre la «description du programme» et sa «documentation auxiliaire». Est inclus de surcroît, dans les dispositions de la directive comme dans le Code français de propriété intellectuelle (article L 112-2 13° du CPI), «le matériel de conception préparatoire[2]».

Le logiciel est qualifié d'œuvre de l'esprit, et à ce titre il est protégeable par le droit d'auteur (voir supra : Chapitre préliminaire).

Distinction entre logiciel spécifique/progiciel :

– *logiciel spécifique* : oeuvre de commande du client au prestataire en vue de la conception et de la réalisation d'un programme adapté à ses besoins ;
– *progiciel* : «ensemble complet et documenté de programmes conçu pour être fourni à plusieurs utilisateurs en vue d'une même application ou d'une même fonction» (Arr. Min. sur l'enrichissement du vocabulaire informatique, 22 déc. 1981, JO 17 janv. 1982, p. 624).

10.1.3 Open Source.

L'open Source Initiative propose huit critères pour identifier les licences libres qui sont décrits dans le document Open Source Definition[3] :

1. la licence ne doit pas prévoir de redevance en échange de l'accès au logiciel, mais ne doit pas non plus interdire qu'une redevance puisse être exigée,
2. la licence doit prévoir la possibilité d'accéder au code source,
3. la licence doit offrir les libertés de modifier et de distribuer les travaux dérivés en vertu des mêmes conditions que la licence du logiciel initial,
4. la licence, tout en encourageant les modifications et donc l'évolution des programmes, doit garantir la paternité de l'auteur, pour cela, elle peut prévoir que les modifications du logiciel ne pourront être distribuées que sous la forme de «patch files» (fichiers correctifs). Par ailleurs, la licence doit expressément permettre la distribution du logiciel combiné provenant du code source modifié, enfin, la licence peut requérir que les travaux dérivés portent des noms ou des numéros de versions distincts du logiciel original,
5. la licence ne doit pas permettre de discrimination contre des personnes, des groupes ou des contrats dans certains domaines d'activités (notamment commerciaux),

[1]Dir. Cons. CE n°91/250, 14 mai 1991, JOCE 17 mai, n°L122, p. 42.
[2]M. Vivant, «Les créations immatérielles et le droit», sous la direction de M. Vivant [1997], Ellipse, p. 51.
[3]http://www.gnu.org/philosophy/free-sw.html

6. la distribution du logiciel ne doit pas être soumise à des exigences supplémentaires tel qu'un accord de non-révélation,
7. l'autorisation ne doit pas être limitée à un usage particulier,
8. il n'est pas nécessaire que les logiciels qui accompagnent celui soumis à la licence soient également open source.

10.1.4 Copyleft.

La Free Software Fundation considère qu'une licence est libre dès lors qu'elle offre à l'utilisateur les libertés suivantes[4] :

– la liberté d'exécuter le programme pour tous les usages,
– la liberté d'étudier le fonctionnement du programme et de l'adapter à ses besoins et pour cela l'accès au code source est une condition requise,
– la liberté de redistribuer des copies,
– la liberté d'améliorer le programme et de publier ses améliorations et pour cela l'accès au code source est, à nouveau, une condition requise.

10.1.5 Licences de «libres copies».

Les logiciels dits en «libre copie» sont soumis à des licences qui ne permettent pas de les modifier. Aussi, les licences de «libres copies» ne sont pas, à proprement parler, des licences libres.

Licence freeware.

Elle permet au public de disposer gratuitement du logiciel.

Licence shareware.

Il s'agit d'un contrat par lequel l'auteur d'un progiciel en autorise la reproduction à titre gratuit afin de le diffuser et de permettre à tous les utilisateurs de l'essayer avant de l'acheter. L'utilisation est à durée déterminée au terme de laquelle l'utilisateur peut décider de conserver le logiciel en contrepartie du versement d'une redevance.

Licence crippleware.

Elle autorise l'utilisation d'un logiciel sous une version qui ne sera fonctionnelle que le temps de la période d'essai.

10.1.6 Certifi cation des licences libres.

La certification consiste, en l'espèce, à déterminer si une licence est libre ou pas.

La Free Software Fundation (FSF) et l'Open Source Initiative (OSI) sont deux organisations reconnues par les communautés du libre pour valider les licences libres. Ces organisations ont défini chacune des critères de certification :

– pour l'Open Source Initiative (voir supra 1.3),

[4] http://www.gnu.org/philosophy/free-sw.html

– pour la Free Software Fundation (voir supra 1.4).

Open Source Initiative.

L'OSI a mis en place une procédure de certification de licences pour déterminer si une licence peut être déclarée officiellement une licence Open Source[5].

Pour soumettre une licence au processus de validation, il faut écrire à l'adresse license-approval@opensource.org.

Une fois la licence approuvée, il est permis de dire que le logiciel qui est sous cette licence est "OSI Certified Open Source Software".

Free Software Fundation.

La FSF a un système un peu moins formel. Elle maintient une page listant des licences avec un commentaire pour chacune d'elles[6]. La FSF classifie les licences en fonction des points clefs suivants :

– la licence est-elle une licence logiciel libre ?
– la licence est-elle de type copyleft ou non ?
– la licence est-elle compatible avec la GNU GPL (c'est-à-dire peut-on combiner un module sous GNU GPL et un autre sous une autre licence pour en faire un module plus grand) ?
– la licence pose-t-elle des problèmes pratiques particuliers ?

La FSF possède une liste permettant d'évaluer toute licence, et de répondre à toute question concernant notamment la GNU GPL[7].

10.2 Analyse générale.

Les licences de logiciels libres ont des caractéristiques générales qui ne manquent pas de soulever des interrogations juridiques.

10.2.1 La langue.

Présentation.

Il n'existe pas aujourd'hui de licence de logiciel libre en langue française, elles sont toutes rédigées en anglais. Cela tient essentiellement à leur origine (voir l'étude relative aux usages des licences libres d'Yves Rougy). On peut également penser que le choix d'une langue internationale telle que l'anglais est le mieux adapté pour la distribution d'un logiciel qui se veut sans frontière. Si le contexte de leur création et les contingences pratiques justifient ce choix, est-il pour autant valable au regard du droit français ?

L'analyse juridique.

Les textes principaux relatifs à l'emploi obligatoire de la langue française sont les suivants :

– loi n° 94-665 du 4 août 1994 relative à l'emploi de la langue française (J.O. 5 avril 1994) dite «loi Toubon» qui supprime et remplace la loi du 31 décembre 1975 n° 75-1349 (J.O. 4 janvier 1976) ;

[5] http://www.opensource.org/licenses/index.html
[6] http://www.gnu.org/licenses/license-list.html
[7] licensing@gnu.org

– décision du Conseil Constitutionnel n° 94-345 du 29 juillet 1994 (J.O. du 2 août 1994) ;
– décret n° 95240 du 3 mars 1995 pris pour l'application de la loi n°94-665 du 4 août 1994 relative à l'emploi de la langue française (J.O. du 5 mars 1995) ;
– circulaire du 19 mars 1996 concernant l'application de la loi n°94-665 du 4 août 1994 relative à l'emploi de la langue française (J.O. du 20 mars 1996) ;
– circulaire du 6 mars 1997 relative à l'emploi du français dans les systèmes d'information et de communication des administrations et établissements publics de l'Etat (J.O. du 20 mars 1997).

L'emploi de la langue française est obligatoire dans :

– «la désignation, l'offre, la présentation, le mode d'emploi ou d'utilisation, la description de l'étendue et des conditions de garantie d'un bien, d'un produit ou d'un service, ainsi que dans les factures et quittances»[8] ;
– cette obligation «n'est pas applicable à la dénomination des produits et spécialités d'appellation étrangère connus du plus large public»[9] ;
– les contrats «quels qu'en soient l'objet ou la forme» «auxquels une personne morale de droit public ou une personne privée exécutant une mission de service public sont parties»[10].

Le champ d'application de la loi concerne les personnes privées comme les personnes publiques. Toutefois, certaines de ces dispositions sont plus contraignantes pour les personnes de droit public et les personnes privées exécutant une mission de service public[11].

L'emploi de la langue française dans les contrats ne semble obligatoire que lorsqu'une personne morale de droit public ou une personne privée exécutant une mission de service public est partie. De plus, les dispositions de l'article 2 de la loi n° 94-665 du 4 août 1994 semblent avoir vocation à s'appliquer qu'en cas de «commercialisation des biens, produits et services»[12]. Enfin, ne sont pas concernés par cette obligation «les factures et autres documents échangés entre professionnels, personnes de droit privé françaises et étrangères, qui ne sont pas consommateurs ou utilisateurs finaux des biens, produits ou services»[13].

Les sanctions sont énumérées dans le Décret n°95240 du 3 mars 1995. Le non-respect des dispositions de la loi est puni d'une peine d'amende prévue pour les contraventions de IVe classe.

Ainsi, l'obligation d'employer la langue française n'est pas générale. Ce sont essentiellement les personnes publiques qui sont concernées.

10.2.2 L'obligation d'un dépôt légal.

Les textes principaux :
– Loi n°92-546 du 20 juin 1992 relative au dépôt légal (J.O. 23 juin 1992) ;
– Décret n°93-1429 du 31 décembre 1993 relatif au dépôt légal (J.O. du 1 janvier 1994).

Le dépôt légal est exigé :

– pour «les progiciels, les bases de données, les systèmes experts et les autres produits de l'intelligence artificielle sont soumis à l'obligation légale dès lors qu'ils sont mis à la disposition du public par la diffusion d'un support matériel, quelle que soit la nature de ce support»[14]. La forme de diffusion généralement adoptée des logiciels libres est la diffusion en ligne. Ainsi l'exigence de dépôt légal ne sera pas applicable dans la plupart des cas. Cependant, il arrive que le logiciel libre soit diffusé sous la

[8] Article 2 al 1 de la loi n° 94-665 du 4 août 1994.
[9] Article 2 al 2 de la loi n° 94-665 du 4 août 1994.
[10] Article 5 de la loi n° 94-665 du 4 août 1994.
[11] Article 2 de la ciculaire du 19 mars 1996.
[12] Article 2.1 de la ciculaire du 19 mars 1996
[13] Article 2.1, §2 de la ciculaire du 19 mars 1996.
[14] Article 1 de la Loi n°92-546 du 20 juin 1992.

forme d'un CD-Rom payant avec notamment une notice complète et un contrat de[15]. Dans ce cas, les personnes qui édictent, ou en l'absence d'éditeur, celles qui produisent ou importent les progiciels sont astreintes au dépôt[16]. Il est à noter que dans un rapport, le Conseil d'État préconise de modifier la loi afin d'étendre l'obligation au cas de mise à disposition du public par le réseau numérique «ouvert»[17] ;

– pour les progiciels, les bases de données, les systèmes experts et les autres produits de l'intelligence artificielle jugés représentatifs par décision des ministres chargés de la Culture de l'Industrie et de la Recherche sur proposition de la commission consultative et après avis du conseil scientifique du dépôt légal. Ces arrêtés sont publiés au Journal Officiel. Le dépôt doit être fait dans un délai de huit jours à compter de la date de publication au Journal Officiel. L'absence de dépôt n'affecte pas la protection par le droit d'auteur, mais elle est sanctionnée pénalement par une amende[18].

L'obligation en l'espèce ne concerne, par conséquent, qu'un nombre limité de logiciel.

10.2.3 La version.

Présentation.

Les licences libres les plus élaborées présentent une clause qui prévoit que de nouvelles versions de la licence pourront être éditées par la suite (exemples : Mozilla Public License version 1 et Netscape Public License, article 6 ; IBM public License version 1.0, article 7 & 4 ; General Public License version 2, article 9 ; Apple Public Source License version 1.2, article 7 ; Sun Community Source License, version 3, article V. A). Seule l'organisation à l'origine de la licence est habilitée à créer une nouvelle version qui aura un numéro distinct.

Exemples :

L'Apple publique source license : 7. Versions of the License. "Apple may publish revised and/or new versions of this License from time to time. Each version will be given a distinguishing version number. Once Original Code has been published under a particular version of this License, You may continue to use it under the terms of that version. You may also choose to use such Original Code under the terms of any subsequent version of this License published by Apple. No one other than Apple has the right to modify the terms applicable to Covered Code created under this License."

Netscape Publique license : 6. Versions of the License. "6.1. New Versions. Netscape Communications Corporation ("Netscape") may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number. 6.2. Effect of New Versions. Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License published by Netscape. No one other than Netscape has the right to modify the terms applicable to Covered Code created under this License."
La clause est identique dans la licence Mozilla.

IBM Publique License : article 7. "IBM may publish new versions (including revisions) of this Agreement from time to time. Each new version of the Agreement will be given a distinguishing version number. The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. No one other than IBM has the right to modify this Agreement."

[15] Les sociétés RedHat, MandrakeSoft, Caldera ou SuSE distribuent ainsi les différentes versions de Linux.
[16] Article 4.3°. de la loi de 1992.
[17] «Internet et les réseaux numériques», les études du Les études du Conseil d'État [1998], La Documentation française.
[18] Article 7 loi 1992.

Les effets prévus de l'apparition d'une nouvelle version.

En général, les licences prévoient que l'utilisateur a le choix entre se prévaloir des termes soit de la nouvelle version soit de la version en vigueur lorsqu'il a commencé à utiliser le logiciel.

La General Public License version 2, article 9 distingue selon les hypothèses : Si le logiciel indique un numéro de version ainsi que la mention «toute version ultérieure», alors, l'utilisateur peut se prévaloir des dispositions de la version indiquée ou de toute autre version plus récente publiée par la Free Software Fundation. Si le logiciel ne précise aucun numéro de version, l'utilisateur peut choisir l'une quelconque des versions publiées par la Free Software Fundation.

L'analyse juridique.

Une des parties ne peut pas prévoir valablement de modifier unilatéralement les termes d'un accord quand bien même les modifications apportées seraient mineures et ne modifieraient pas l'esprit de la licence[19]. En effet, le consentement de l'utilisateur a été donné pour les conditions prévues par la licence dont il a eu connaissance.

Dans l'hypothèse de ces licences, la modification de l'accord n'est pas, à proprement parler, unilatérale dans la mesure où il appartient à l'utilisateur d'accepter la nouvelle version ou, au contraire, de la refuser et rester soumis à la version initiale. Ainsi, pour être opposable au licencié, la modification doit faire l'objet d'un accord entre les parties. Aussi, ces clauses sont juridiquement acceptables.

Enfin, il n'est pas inutile de remarquer que de nouvelles versions sont assez rares. Les licences de logiciels libres sont, somme toute, stables.

10.2.4 Les clauses relatives à la garantie et à la responsabilité.

Toutes les licences de logiciels libres contiennent une clause élusive de responsabilité.

Les exemples.

Mozilla Public License version 1 et Nescape Public License.

"7. DISCLAIMER OF WARRANTY

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED CODE IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED CODE IS WITH YOU. SHOULD ANY COVERED CODE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER."

"9. LIMITATION OF LIABILITY

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL,

[19] La GPL, par exemple, précise que les nouvelles versions seront analogues les unes par rapport aux autres. Autrement dit d'une version à l'autre, l'esprit reste le même, seuls diffèrent quelques détails afin de résoudre de nouveaux problèmes ou de nouvelles situations.

INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THAT EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU."

IBM public License version 1.0.

"5. NO WARRANTY EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement, including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations."

Voir également Apple Public Source License version 1.2, article 8 ET 9 ; Sun Community Source License, version 3, article V B et C, ...

L'analyse juridique.

Peut-on ainsi valablement écarter toute garantie et toute responsabilité ?

Cette question n'est pas nouvelle, elle est antérieure au développement des licences libres. La Commission des clauses abusives a considéré le problème suffisamment préoccupant pour intervenir en émettant une recommandation le 7 avril 1995 «relative aux contrats proposés par les éditeurs ou distributeurs de logiciels ou progiciels destinés à l'utilisation sur micro-ordinateurs»[20].

À cette occasion La Commission relève huit clauses contractuelles jugées abusives au regard de la loi dont celles relatives à la garantie. En effet, elle constate que «ces contrats sont élaborés par les seuls professionnels et que l'utilisation de ces logiciels emporte généralement adhésion des consommateurs à leurs clauses(...) ils constituent ainsi des contrats habituellement proposés par des professionnels à des consommateurs». Les règles du droit de la consommation leurs sont donc applicables. Elle recommande, notamment, «que soit éliminées des contrats objets de la présente recommandation les clauses qui ont pour objet ou pour effet : (...)

1° d'exclure toute garantie du professionnel afférente au logiciel, à son support et de l'exonérer de toutes les conséquences des défauts de la documentation fournie lors de la mise à disposition du logiciel ;

2° d'induire en erreur le consommateur en combinant des stipulations qui excluent toute garantie avec des clauses limitatives de garantie ;

3° d'exonérer le professionnel de toute responsabilité du fait des conséquences dommageables de l'utilisation des logiciels qu'il commercialise.»(...)

[20]Recommandation n°9502, BOCCRF 25 août 1995.

Ce texte en tant que recommandation a une portée limitée, toutefois il présente une force incitatrice et peut servir de référence aux juges. On peut se demander si toutes les licences de logiciels libres relèvent de ce texte dans la mesure où la Commission ne vise que les contrats élaborés par des professionnels et que les logiciels libres ne sont pas tous communiqués par des professionnels. Mais surtout, il ne fait aucun doute que les contrats visés sont ceux utilisés dans la commercialisation des logiciels mêmes. En 1995, les nouveaux modèles économiques tels que ceux qui font l'objet de ce rapport RNTL n'étaient pas encore émergents et les licences libres n'étaient connues que des initiés. Or ce nouveau type de licences n'organise pas le même rapport de force entre le distributeur et l'utilisateur. En effet, il convient de se demander si le caractère gratuit des licences libres mais surtout si l'étendue des libertés offertes à l'utilisateur, ne doit pas conduire à d'autres conclusions que celles retenues pour les licences classiques. La législation pour la protection du consommateur tend à maintenir un équilibre entre les parties en venant en aide aux consommateurs considérés comme la partie faible. Or, si la Commission a pu considérer que les clauses élusives de responsabilité dans les licences classiques créent un déséquilibre significatif , l'équilibre est bien différent s'agissant des licences libres. Ainsi, l'absence de garantie pourrait se justifier dans les licences libres, nous semble t-il, par l'importance des libertés accordées à l'utilisateur, notamment celle de modifier le logiciel, et ceci à titre gratuit.

Néanmoins et nonobstant ce qui vient d'être affirmé, la loi du 19 mai 1998[21] transposant la directive sur les produits défecteux[22] fait peser sur le fabricant (article 1386-6 C.Civ.) une obligation de réparation des atteintes à la personne ou à un bien autre que le produit défectueux (article 1386-2 C. Civ). Il découle de ces dispositions qu'il y a un principe général de responsabilité du fabricant pour le défaut des produits qu'il met en circulation, et que le droit à réparation bénéficie aussi bien aux tiers qu'aux acquéreurs de tels produits. Ces solutions valent pour l'ensemble de l'Union Européenne. La définition du produit visé par la loi est large (article 1386-3 C. Civ.), de ce fait, la notion semble englober les logiciels dès lors que ces derniers apparaissent comme défectueux (article 1386-4 C. Civ.). Pour obtenir réparation, il suffit de prouver «le dommage, le défaut et le lien de causalité entre le défaut et le dommage» (article 1386-9 C. Civ.). Et un «produit est défectueux (...) lorsqu'il n'offre pas la sécurité à laquelle on peut légitimement s'attendre» (article 1386-4). Cette responsabilité ne s'éteint que «dix ans après la mise en circulation du produit» qui a causé le dommage (article 1386-16). La loi étend le régime de responsabilité du fabricant au loueur ainsi qu'à tous les autres «fournisseurs professionnels» (article 1386-7 C. Civ.). Ces dispositions pourraient s'appliquer au donneur de licence, dès lors qu'il s'agit d'un fournisseur professionnel. Ainsi, le développeur indépendant dont la profession n'est pas de fournir des logiciels, ne devrait pas être inquiété par cette lourde responsabilité. Par contre, le fournisseur professionnel n'a pas la possibilité de s'exonérer de sa responsabilité. Toute clause contraire est réputée non écrite sauf si la clause est stipulée entre professionnels ou que le dommage a été causé à un bien dont l'usage ou la consommation n'est pas principalement privée (1386-15 C. Civ.), il ne peut en effet se prévaloir dans cette hypothèse que des cas d'exonération exceptionnels énumérés par l'article 1386-11 du Code Civil.

10.3 Essai de classifi cation des licences de logiciels libres.

Il est devenu classique, aujourd'hui, de différencier trois types de licences de logiciels libres : les licences dites de domaine public, les licences des firmes commerciales et les licences copyleft. Cette classification repose principalement sur les effets des licences et secondairement sur leur origine.

[21] Loi n°98-389, 19 mai 1998, JO 21 mai 1998.

[22] Directive n°85/374/CEE, Cons. CE relative au rapprochement des dispositions législatives, réglementaires et administratives des Etats membres en matière de responsabilité du fait des produits défectueux, [JOCE, n°L210, 7 août 1985.]

10.3.1 Les licences dites de domaine public.

Les plus connues sont la licence BSD de Berkeley, la licence Apache (Apache Sofware License), la licence pour Mozart (License Agreement for Mozart), la licence Cryptix (Cryptix General License), et la licence de MIT. Ces licences ont de commun d'être très courtes et d'avoir un ton déclaratif. Trois points sont traités par ces licences : les utilisations autorisées, les conditions d'utilisation et la garantie.

Les utilisations autorisées.

Les utilisations autorisées sont celles habituelles des licences libres :

- l'autorisation d'utiliser le logiciel ;
- l'autorisation de modifier le logiciel ;
- l'autorisation de distribuer le logiciel modifié ou non.

Les conditions d'utilisation.

Les conditions d'utilisation de ces licences libres sont peu contraignantes.

La mention du copyright.

L'exemple de l'Apache Sofware License :

- la distribution du code source doit conserver la mention de copyright, la liste des conditions de chaque autorisation et la décharge de responsabilité ;
- la distribution de la forme binaire du logiciel doit reproduire ces mêmes indications dans la documentation et/ou les autres matériaux fournis avec la distribution ;
- s'il y a une documentation attachée à la distribution, il convient d'indiquer la formule suivante : "This product includes sofware developed by [nom de l'organisme] (http ://www.apache.org/)" ;
- cette reconnaissance peut apparaître dans le logiciel lui-même, et partout où tels remerciements de tiers apparaissent normalement.

La protection du nom de l'organisme à l'origine de la licence.

Ces licences sont rédigées par des organismes, comme l'Université de Berkeley ou le MIT, pour accompagner les logiciels qu'ils développent en interne. Elles n'ont pas vocation à être utilisées pour les besoins de tiers qui souhaiteraient diffuser leurs propres logiciels. Toutefois, des développeurs peuvent s'en inspirer afin de rédiger leur licence. On peut d'ailleurs constater que la pratique consistant à reprendre quasiment mot à mot les licences existantes en se contentant de changer le nom est très courante et n'en demeure pas moins contestable au regard du droit d'auteur. En effet, dès lors qu'une licence est originale le droit protège l'auteur des contrefaçons...

Le fait que ces licences soient rédigées pour les besoins propres aux détenteurs des droits (qui sont généralement des personnes morales) fait que leur nom est présent dans le titre de la licence et que leur copyright est indiqué(par exemple : "Copyright (c) 2000 The Apache Sofware Fundation. All rights reserved".)

Si l'utilisation du logiciel soumis à une telle licence est libre, le nom de l'organisme ne doit pas servir à des fins promotionnelles. Les organismes entendent ainsi protéger leur nom et leur réputation.

La clause de non garantie.

Nous ne reviendrons sur le sujet déjà abordé (voir supra 2.4).

10.3.2 Les licences des firmes commerciales.

Certaines sociétés de distribution de logiciels ont décidé de développer, en parallèle, des logiciels sous une forme plus ou moins libre. Ces firmes ont rédigé leurs propres licences. Mais, ce phénomène reste isolé.

Le choix des sociétés d'utiliser leurs propres licences ne s'explique pas uniquement par le souci d'y retrouver leurs clauses types, mais également, par la volonté de ne pas nuire à leur activité commerciale de distribution de logiciels qu'elles n'ont pas abandonnée.

Une rédaction classique.

Les licences des firmes commerciales utilisent des licences dont la rédaction est classique dans la mesure où l'on y retrouve les clauses standards des contrats :

- durée du contrat,
- clause désignant la loi applicable au contrat,
- clause indiquant la juridiction compétente en cas de litige...

La présence de ces clauses n'a pas, a priori, à émouvoir tant elles sont classiques. Pourtant, lorsqu'il s'agit de licences libres, leur présence peut surprendre car, dans cette hypothèse, elles sont rares. Elles se trouvent uniquement dans les licences des firmes commerciales.

On peut imaginer plusieurs raisons à ce phénomène : Les rédacteurs des licences libres des firmes commerciales sont les mêmes que ceux qui rédigent les autres licences dites propriétaires. Ils reproduisent ce qui ont l'habitude de faire en appliquant leurs modèles. D'autre part, si ces clauses sont fortes utiles, lorsque l'on connaît l'identité du donneur de licence comme pour les licences des firmes commerciales, elles seront bien souvent sans portée dans les autres cas. Ainsi, s'agissant d'une licence comme la GNU GPL, qui peut être utilisée par tout auteur d'un logiciel quelle que soit sa nationalité ou son lieu de résidence, le fait d'indiquer la loi applicable ou la juridiction compétente sera bien souvent sans effet.

La conciliation d'une distribution libre et d'une distribution propriétaire.

Afin de concilier leurs deux modes de distribution de logiciels, l'un aux conditions de licences libres et l'autre aux conditions propriétaires, les firmes commerciales se réservent fréquemment certaines prérogatives face aux utilisateurs. De plus, elles circonscrivent les libertés accordées uniquement aux droits attachés aux logiciels à l'exclusion de leurs autres droits intellectuels relatifs notamment à une marque ou un logo. De telles dispositions protectionnistes sont parfois perçues comme contraires à la philosophie du libre. Aussi, pour entériner l'image libre de certains projets, les firmes commerciales ont parfois adopté une licence distincte qui ne contient pas de telles restrictions.

La réservation de prérogatives.

La Netscape Public License version 1.0 présente un très bon exemple quant à la réservation de certaines prérogatives en vue de concilier la distribution libre avec ses distributions propriétaires. La licence organise un régime particulier au profit de Netscape. En effet, si en principe les modifications apportées au logiciel sont régies par les conditions de la licence (article 3.1), la société Netscape peut, si elle le désire, échapper à ces obligations : elle se réserve le droit d'inclure le logiciel dans un de ses produits non libres pendant les deux années suivant la date de la version de code initial et que l'ensemble ne soit pas soumis à la licence.

La société Apple, pour sa part, s'est réservée la liberté de choisir si les modifications qu'elle fait ou qui sont faites pour son compte seront ou non soumises aux conditions de la licence publique d'Apple ou à des conditions différentes. Il est permis de se demander si cette option est offerte également aux autres contributeurs qui auraient apporté des modifications au logiciel initial. La clause est peu claire sur le sujet :

dans un premier temps, il est précisé que les contributeurs conservent leurs droits sur leurs contributions ainsi qu'Apple et, dans un second temps, elle précise la liberté d'Apple dans le choix de distribution des modifications sans viser les autres contributeurs.

Il semble probable qu'Apple n'ait pas souhaité étendre cette liberté sans pour autant afficher clairement ce choix : "11. Ownership. Subject to the licenses granted under this License, each Contributor retains all rights, title and interest in and to any Modifications made by such Contributor. Apple retains all rights, title and interest in and to the Original Code and any Modifications made by or on behalf of Apple ("Apple Modifications"), and such Apple Modifications will not be automatically subject to this License. Apple may, at its sole discretion, choose to license such Apple Modifications under this License, or on different terms from those contained in this License or may choose not to license them at all."

L'exclusion des autres droits de propriété intellectuelle.

Les licences des firmes commerciales précisent qu'elles ne confèrent aucun droit sur le logo et les marques de la firme :

– la licence publique de Netscape : III. Netscape and logo. This License does not grant any rights to use the trademark "Netscape", the "Netscape N and horizon" logo or the Netscape lighthouse logo, even if such marks are included in the Original Code ;
– IBM Public License Version 1.0 : "Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppels or otherwise. All rights in the Program not expressly granted under this Agreement are reserved" ;
– la licence publique d'Apple : 10. "Trademarks. This License does not grant any rights to use the trademarks or trade names "Apple", "Apple Computer", "Mac OS X", "Mac OS X Server", "QuickTime", "QuickTime Streaming Server" or any other trademarks or trade names belonging to Apple (collectively "Apple Marks") or to any trademark or trade name belonging to any Contributor. No Apple Marks may be used to endorse or promote products derived from the Original Code other than as permitted by and in strict compliance at all times with Apple's third party trademark usage guidelines which are posted at http://www.apple.com/legal/guidelinesfor3rdparties.html".

Le choix d'une licence différente pour chaque démarche.

Certaines entreprises ont préféré créer une licence plus neutre pour des projets indépendants dont elles étaient l'initiatrice.

Ainsi Netscape dispose de deux licences pour ses projets libres :
– la «Netscape Publique license» ;
– la «Mozilla Publique license».

Sun a opté pour la même stratégie :
– la «Sun Community Source License» ;
– la «Sun Public License».

10.3.3 Les licences «copyleft».

Il s'agit des licences créées sur l'initiative de Richard M. Stallman : la Licence Publique Générale (la GNU GPL), et la GNU licence publique générale amoindrie (la GNU LGPL).

Ces licences offrent les libertés habituelles des licences libres, à savoir, les libertés de copier, de modifier et de distribuer le Logiciel[23] modifié ou non. Par ailleurs, elles organisent l'accès au code source au profit des

[23]Le terme «Logiciel» avec une majuscule désigne, dans ce développement, le logiciel auquel s'applique la licence. La GPL s'applique non seulement au «Logiciel» , mais également à «l'ouvrage fondé sur le Logiciel», c'est-à-dire l'ouvrage contenant le

utilisateurs et précisent qu'il est permis également de facturer l'acte physique de transfert et de copie d'un exemplaire, et de proposer une garantie en échange d'une rémunération.

L'originalité de ces licences réside essentiellement dans les conditions d'exercice des libertés annoncées. En effet, la GNU GPL a été rédigée dans un souci de préservation des libertés de l'utilisateur. Ceci se traduit par des contraintes parfois considérées comme trop lourdes. C'est pourquoi, à côté de la GNU GPL, a été prévue la GNU LGPL qui ne comporte pas ces mêmes contraintes mais qui, par conséquent, ne préserve pas autant la liberté d'utilisation, d'où son nom : la licence publique générale «amoindrie».

Les conditions d'exercice des libertés accordées par la GNU GPL.

La GPL permet aux utilisateurs de copier et de diffuser le logiciel à condition de :

– apposer, d'une manière parfaitement visible et appropriée, sur chaque exemplaire, une mention de droits d'auteur (Copyright), et une clause limitative de garantie ;
– maintenir intacts tous les avis qui se rapportent à la licence et à l'absence de toute garantie ;
– transmettre à tout destinataire du Logiciel un exemplaire de la GPL en même temps que le Logiciel.

La GNU GPL permet de modifier le Logiciel et de diffuser le Logiciel modifié à condition de :

– ajouter aux fichiers modifiés l'indication très claire des modifications effectuées, ainsi que la date de

---

Logiciel ou une version de ce dernier contenant ou non des modifi cations et/ou traduit en une autre langue (la GPL précise, en effet, que la traduction est considérée comme une modifi cation).

Tout élément qui peut être identifi é et qui est indépendant du Logiciel n'est pas soumis à la licence. Dès lors la licence ne s'applique à de tels éléments distribués seuls. Mais quand ces mêmes éléments sont distribués en tant qu'éléments d'un tout qui représente un ouvrage se fondant sur le Logiciel, la distribution de ce tout doit se faire conformément aux dispositions de la GPL.

En outre, l'assemblage d'un travail, non fondé sur le Logiciel, avec le Logiciel (ou un travail dérivé de ce dernier), effectué sur un support de stockage ou de distribution, ne fait pas entrer ce travail dans le champ d'application de la GPL. Aussi tant que l'on ne peut pas constater d'intégration de code, les développements indépendants du Logiciel n'entrent pas dans le champ d'application de la licence.

S'agissant de la LGPL, un programme, qui est seulement conçu pour fonctionner avec la Bibliothèque, sort du champ d'application de la licence. Et en principe, quand tout ou portion du code de la Bibliothèque est intégré dans un ensemble logiciel, cet ensemble tombe sous l'application de la LGPL.

Toutefois, le produit, résultant de la combinaison de la Bibliothèque avec un programme qui est conçu pour fonctionner avec cette Bibliothèque, peut échapper au champ d'application de la licence alors même qu'il contient des portions de la Bibliothèque. Dans cette hypothèse, si la LGPL ne s'applique pas au produit, il faut néanmoins que la licence à laquelle il est soumis, autorise sa modifi cation pour un usage personnel et permette de procéder à de l'ingenierie inverse, afi n de corriger les erreurs. De plus, certaines conditions particulières sont requises lors de la diffusion de ce travail :

– une note claire doit indiquer qu'une bibliothèque soumise à la LGPL a été utilisée pour la conception de ce travail,
– une copie de la licence doit être fournie,
– les copyrights de la bibliothèque ainsi que des explications pour trouver une copie de la licence doivent être inclus au côté des copyrights du produit si ces derniers s'affi chent,
– l'intégralité du code source de la bibliothèque doit être transmis,
– enfi n, il faut respecter l'un des points suivants :

1. l'intégralité du code source de la bibliothèque doit accompagner le travail. Lorsque le produit est un exécutable lié à la bibliothèque, il faut proposer l'ensemble du produit sous la forme de code source ou objet de telle sorte que l'utilisateur puisse modifi er la bibliothèque.

2. il faut une offre écrite, valable pendant au moins 3 ans, proposant pour un tarif n'excédent pas le coût de la copie soit la fourniture de l'intégralité du code source de la bibliothèque, soit la fourniture du produit sous la forme de code source lorsque le produit est un exécutable lié à la bibliothèque.

3. les sources doivent être exploitables, autrement dit, elles doivent être accompagnées des données et des outils (autres que ceux que l'on trouve dans les distributions de système d'exploitation) afi n de permettre de générer l'exécutable.

4. si la distribution se fait par un système d'accès à une copie située à un endroit désigné, alors, il faudra qu'au même endroit, on y trouve les mêmes éléments ci-dessus décrits (notamment la version sous la forme de code source de la bibliothèque),

5. il convient de vérifi er que les utilisateurs ont reçu les codes source ou la proposition de les obtenir, conformément à ce qui vient d'être dit.

chaque changement. Cette obligation est motivée par la nécessité d'avertir les destinataires de la copie, qu'il ne s'agit pas de l'original, afin qu'une copie défectueuse n'entache pas la réputation de l'auteur du Logiciel original,
– distribuer, sous les termes de la Licence, l'ensemble de toute réalisation contenant tout ou partie du programme, avec ou sans modification. Ainsi, il est permis de modifier le logiciel à condition d'offrir sur le Logiciel modifié les mêmes droits contenus dans la GPL (et qui en outre ont permis de procéder à la modification),
– si le programme modifié lit normalement des instructions interactives lors de son exécution, l'utilisateur doit faire en sorte, lorsqu'il commence l'exécution du programme pour une telle utilisation interactive, que ce programme imprime ou affiche une annonce de la manière la plus usuelle, comprenant :
  – un avis approprié de droits d'auteur,
  – et un avis selon lequel il n'y a aucune garantie (à moins de fournir de son propre chef une garantie). De plus, cet avis doit indiquer aux utilisateurs d'une part qu'ils peuvent redistribuer le programme au titre des dispositions de la GPL, et d'autre part comment ils peuvent visualiser une copie de la GPL.

Cette obligation ne s'impose pas si le programme par lui-même est interactif et qu'il n'imprime normalement pas une telle annonce. Dans cette hypothèse, l'ouvrage se fondant sur le programme n'a pas besoin d'imprimer une annonce.

Afin de permettre ces modifications, l'accès au code source est garanti.

La GNU GPL permet la diffusion du Logiciel modifié ou non, en code objet ou sous une forme exécutable, à condition de :

– fournir le code source complet correspondant, lisible par un système informatique. Code qui doit être distribué sur un support couramment utilisé pour l'échange de logiciel,
– fournir une offre écrite, valable pendant au moins trois ans, prévoyant de donner à tout tiers qui en fera la demande une copie, sous forme lisible par ordinateur, du code source correspondant, pour un tarif n'excédant pas le coût de la copie, sur support couramment utilisé pour l'échange de logiciels,
– fournir les informations reçues, pour proposer de fournir le code source correspondant (cette alternative n'est autorisée que pour la distribution non commerciale et seulement si le concessionnaire a reçu le programme sous forme objet ou exécutable, selon les modalités décrites à l'alinéa b) ci-dessus).

Plusieurs dispositions de la GNU GPL ont pour objet d'empêcher toute appropriation ultérieure du code :
– il est interdit pour l'utilisateur, qu'il ait ou non modifié le Logiciel, de restreindre les droits accordés par la GPL aux personnes auxquelles il transmet des copies modifiées ou non du Logiciel ;
– il n'est par permis d'incorporer le Logiciel ou une partie de ce dernier à d'autres logiciels libres dont les conditions de distributions sont différentes sans l'autorisation de l'auteur ;
– enfin, de manière implicite dans la version 2 et explicite dans la version 1, la licence interdit à l'utilisateur d'incorporer le Logiciel ou une partie de ce dernier à d'autres logiciels non libres.

Les conditions d'exercice des libertés accordées par la GNU LGPL.

La GNU LGPL permet de copier et de diffuser la Librairie à condition de

– apposer, sur chaque exemplaire, une mention de droits d'auteur (Copyright) et une clause limitative de garantie ;
– maintenir intacts tous les avis qui se rapportent à la licence et à l'absence de toute garantie ;
– transmettre à tout destinataire de la bibliothèque un exemplaire de la LGPL en même temps que la librairie.

La GNU LGPL permet de modifier et de diffuser la version modifiée de la Librairie à condition :

– de respecter les mêmes conditions que lorsqu'il diffuse une copie non modifiée de la bibliothèque,
– que le travail dérivé soit lui-même une bibliothèque logicielle,

– que quand la bibliothèque modifiée utilise des informations du programme d'application, sans que les informations soient des paramètres de la fonctionnalité, le résultat reste inchangé en l'absence de ces informations.

Contrairement à la GPL, la LGPL permet d'incorporer des fonctionnalités de la Bibliothèque à une autre bibliothèque soumise à une licence différente de la LGPL et de distribuer la bibliothèque résultante à condition de l'accompagner de :

– la Bibliothèque entière soumise à la LGPL,
– l'indication très claire qu'une portion de la bibliothèque provient d'une Bibliothèque soumis à la LGPL ainsi que l'endroit où l'on peut trouver la version non intégrée de cette Bibliothèque.

Enfin, la GNU LGPL permet de diffuser la Librairie modifiée ou pas, en code objet ou sous forme exécutable, à condition de :

– fournir le code source complet correspondant, lisible par un système informatique, et sur un support couramment utilisé pour l'échange de données. Le code source désigne, selon la licence, la forme sous laquelle les modifications sont les plus aisées ;
– quand la distribution du code objet consiste à offrir un accès permettant de copier la bibliothèque depuis un endroit particulier, fournir un accès équivalent pour se procurer le code source au même endroit.

La compatibilité des licences.

Le principe.

En limitant les cas où il est permis d'incorporer le Logiciel à d'autres logiciels soumis à des licences qui n'offrent pas le même degré de protection des libertés d'utilisation, la GNU GPL devient incompatible avec toutes ces licences. En effet, aucune intégration physique des codes n'est alors permise.

Il convient dès lors de vérifier avant toute incorporation si celle-ci est autorisée. La GNU GPL invite, dans ce cas, l'utilisateur à contacter l'auteur du Logiciel. Par ailleurs, la Free Software Fundation répertorie les licences qu'elle considère compatibles avec la GNU GPL et celles qu'elle considère incompatible. Cette validation (cf. supra 1.6) permet à l'utilisateur de savoir si l'incorporation est possible pour tous les projets GNU qui concernent des logiciels pour lesquels la FSF dispose des droits de propriété intellectuelle. Quant aux autres logiciels, la validation par la FSF est un bon indicateur pour connaître les chances d'obtenir le droit d'incorporer le Logiciel.

Malheureusement, compte tenu de la quantité très importante de licences dites libres, il est courant d'avoir à incorporer un Logiciel à un autre soumis à une licence non répertoriée. Dans ce cas, il n'y a pas d'autre solution que de faire une demande particulière aux auteurs du Logiciel sur la possibilité d'incorporation.

Les palliatifs au problème d'incompatibilité.

Il arrive que les auteurs d'un logiciel modifient leur licence ou l'abandonnent au profit la GPL afin de permettre son incorporation à tous les logiciels soumis à la GPL.

Une autre pratique s'est développée consistant à soumettre un même logiciel à deux licences différentes dont la GPL. Dans ce cas, il est offert à l'utilisateur de choisir entre l'une ou l'autre des licences, le panachage des clauses n'étant pas permis... Ainsi les auteurs ne renoncent pas à leur licence tout en permettant les incorporations avec les logiciels soumis à la GPL.

Pour le cas particulier des bibliothèques, dont l'utilisation suppose à chaque fois une incorporation, la FSF a prévu la GNU LGPL moins contraignante sur le sujet que la GNU GPL. En effet, cette dernière permet d'incorporer des fonctionnalités de la Bibiothèque à une autre bibliothèque soumise à une licence différente de la LGPL et de distribuer la bibliothèque résultante (cf. supra 3.3.2).

Brevets concernant un programme d'ordinateur : l'étendue de la protection.

FRANCK MACREZ,
Université de Montpellier I/ERCIM, Franck.Macrez@free.fr.

Le logiciel, envisagé comme objet de droit, se trouve à la frontière des catégories élémentaires du droit de la propriété intellectuelle, droit d'auteur (œuvre de langage) et brevet (création industrielle).

Instrument de diffusion de la connaissance, le brevet, conçu pour être un moteur d'innovation, permet à celui qui le détient de se prévaloir d'un monopole sur l'utilisation du contenu de l'ensemble formé par les descriptions et revendications de son titre. La contrepartie de ce monopole est la diffusion (sous la forme de publication par l'office de brevet) deds moyens de réalisation de l'invention, permettant sa compréhension et sa reproduction.

La proposition de directive de la Commission européenne du 20 février 2002[1] tente de marquer une étape vers la légalisation de la jurisprudence de l'OEB[2] en matière d'«invention mise en œuvre par ordinateur», malgré la polémique qu'elle engendre notamment sous l'action des acteurs de la communauté du logiciel libre[3].

Le choix avait été fait en 1985 de protéger le logiciel par la loi sur le droit d'auteur, alors que dans le même temps s'amorçait une ouverture dans le droit des brevets avec la célèbre décision Vicom[4].

La question ressurgit depuis quelques années. L'évolution de la jurisprudence des chambres de recours de l'OEB s'explique par l'influence des politiques américaine et japonaise de délivrance de brevets portant sur des logiciels et des méthodes commerciales, ainsi que par la volonté des éditeurs de logiciels de mieux valoriser financièrement leurs produits par un titre industriel. La propriété du brevet permet de le faire figurer dans le bilan comptable, ce qui fût particulièrement intéressant dans le contexte de bulle spéculative des années 1999-2000.

On constate ainsi un détournement de l'institution brevet par la pratique des «brevets de façades», dans lesquels le procédé breveté est la plupart du temps évident (donc facilement annulable). Ils permettraient notamment aux multinationales de faire pression (certains parlent de «petites escroqueries»[5], peut-être plus ou moins petites) sur de petits éditeurs de logiciels afin de leur faire s'acquitter de royalties[6].

Par ailleurs, l'opacité semble être la règle dans ce domaine. La majorité de ces brevets est rarement compréhensible aisément même pour un spécialiste. Les contentieux sont rares, transiger (en secret) est la solution la plus rationnelle pour un entrepreneur en raison des durées de procédures incompatibles avec la durée de vie moyenne d'une technologie logicielle donnée[7]. Et même lorsqu'un contentieux a lieu, le juge se réfère essentiellement au rapport de l'expert, qui n'est pas publié.

Ces titres «de façade» (ou «de barrage») sont en réalité destinés à être annulés lors du premier contentieux porté devant une juridiction judiciaire. Peut-on pour autant affirmer que ces «brevets de logiciel» se résument à ne constituer qu'une arme stratégique économico-financière ? Malgré les difficultés que cette opacité engendre pour celui qui tente d'y voir clair dans la matière, il nous semble fondamental de chercher à définir ce qui est protégé, faire le tri entre ce qui est annulable et ce qui est valable. L'enjeu est de combattre une insécurité juridique omniprésente et qui risque de durer : la proposition de directive de la Commission était attendue depuis plusieurs années (et son apport par rapport à la jurisprudence déjà existante de l'OEB est minime), l'accord avec le Parlement pour le vote effectif d'une directive sera difficile à trouver et, compte tenu de la procédure communautaire, ne devrait pas arriver avant plusieurs années[8].

[1] http://www.europa.eu.int/comm/internal_market/en/indprop/comp/com02-92fr.pdf pour la version française.

[2] Offi ce Européen des Brevets.

[3] Principalement Aful, April, Eurolinux, FFII.

[4] Chambre des Recours Technique (CRT) de l'Offi ce Européen des Brevets(OEB), 15 juillet 1986, VICOM System Inc., PIBD 1987, 409-III, p. 134.

[5] Dominique Foray, rencontre APP, 28/05/02.

[6] Pour la description d'une situation concrète, http://swpat.ffii.org/papiers/eubsa-swpat0202/index.fr.html.

[7] Ibid. : «(...) Vous êtes maintenant une société attractive. Une agence de brevet vous approche : vous violez 2-3 de leurs brevets. Le domaine revendiqué est large. Ils veulent 100.000 €. Une bataille juridique pourrait vous bloquer pendant 10 ans et coûter 1 million d' €. Vous payez. Un mois plus tard le prochain agent de brevet frappe a votre porte ....»

[8] En étant optimistes, nous pouvons évaluer à 2 ans la procédure de «navette» ; à laquelle il faut ajouter un délai de transposition de la directive de 3 à 4 ans.

L'étendue de la protection d'un brevet concernant un programme d'ordinateur peut se déterminer au stade de sa naissance (1) -la délivrance du titre par un office-, ainsi qu'à l'occasion de la mise en œuvre de ses prérogatives par le breveté (2). Il nous paraît important, aussi, de distinguer les deux réalités différentes que recouvre l'expression «invention concernant un programme d'ordinateur» : le programme d'ordinateur est la seule innovation d'un procédé d'ensemble, ou bien d'autres éléments innovants interviennent dans ce procédé.

11.1 La naissance du brevet.

La demande de brevet, définissant l'objet de la réservation dans les descriptions et revendications, doit porter sur une création «technique».

11.1.1 Les descriptions/revendications...

Les revendications définissent l'étendue de la protection recherchée pour l'invention, mais elles s'interprètent au regard des descriptions qui l'accompagnent.

Le mode de rédaction des descriptions.

Un langage spécifique.
La technique de rédaction du brevet est fondamentale pour que la délivrance par l'office ait lieu. L'important, pour contourner l'exclusion «programme d'ordinateur en tant que tel» (v. *infra*), est de matérialiser l'«invention» en accentuant les aspects physiques de la fonction du logiciel : son action au niveau des connexions physiques («signal en Entrée/Sortie») pour chaque périphérique, voire au niveau des composants (les mémoires, le disque dur, la tête de lecture de la disquette par exemple).

Pour se faire délivrer un brevet concernant un logiciel, il est nécessaire d'«utiliser les termes de «procédé», «invention», «système» ou «dispositif», plutôt que «logiciel» ou «programme»[9], de préférer «système de manipulation interactive» à «système de traitement de l'information», «moyens d'initialisation» à «mise à zéro des registres», «appareil» de compression pour un algorithme ayant la même fonction[10], ou «réservation de plusieurs blocs de mémoire pour une variable dans un appareil de mémoire».

La question de la publication des sources.
Le principe est la rédaction en langage naturel, éventuellement accompagné de schémas explicatifs. La description doit être adaptée à la nature de l'invention : «Peuvent en outre figurer en annexe à la fin de la description notamment : 1° de courts extraits de programmes d'ordinateur présentés sous forme de listage rédigés en langage de programmation courants (...)»[11].

Le code source lui-même n'offre pas une description suffisante pour la compréhension d'un procédé, et l'organigramme -comme complément des descriptions en langage naturel- paraît être le mode le plus approprié pour la *description/compréhension* de l'utilisation particulière d'un algorithme.

Cette faculté de dévoiler le code source du programme concerné est en pratique bien peu exercée, ce qui est regrettable tant du point de vue de la reproduction de l'invention que de l'efficacité des recherches d'antériorités (et de la diffusion de la connaissance, objectif originel du système de brevet).

Dans une logique de revendication d'un produit (v. *infra*), la description devrait permettre en premier lieu

[9] Guide Lamy [2000].
[10] Exemples cités par Linant de Bellefonds [2001], p. 9.
[11] Art. R.612-13 CPI.

la *reproduction* de l'«invention»[12] : une législation reconnaissant la validité du brevet logiciel (organisant la brevetabilité du logiciel) devrait imposer la divulgation du code source pour ce qui est revendiqué comme produit

La forme des revendications.

Du procédé au produit.

Si la revendication dans un «brevet de logiciel» sera toujours fonctionnelle, elle peut être de produit[13] ou de procédé[14].

L'INPI[15] et l'OEB[16] acceptent déjà, depuis l'année 2000, des revendications de produit. Surprenant à première vue (le logiciel est un procédé de traitement de l'information), ce choix s'explique par le «besoin» des titulaires de brevet de se voir faciliter les procédures de saisie-contrefaçon. Prouver l'utilisation d'un procédé (dans les locaux d'une société saisie) est beaucoup plus long que de saisir directement une machine, voire simplement un disque dur, ou tout autre support.

C'est pourquoi des brevets portant sur un produit-programme[17] revendiqué en tant que support, selon une formulation de type : «produit programme d'ordinateur comprenant des portions/moyens/instructions de code de programme enregistré sur un support utilisable dans un ordinateur (...)»[18] ont été favorablement reçus.

Mais la Commission revient en arrière sur ce point en considérant que cela reviendrait à breveter des programmes «en tant que tels»[19] : seul un «produit-machine» pourra être revendiqué[20].

La méthode d'appréciation des revendications.

Le logiciel est un procédé de traitement de l'information ; il peut être revendiqué comme simple étape d'un procédé industriel plus large[21] La méthode de l'appréciation de l'invention dans son ensemble[22] adoptée par l'OEB -par opposition à celle consistant à s'attacher au «noyau essentiel»[23] de l'invention- permet de revendiquer la combinaison du logiciel et du matériel alors que l'unique élément nouveau et inventif peut être le logiciel. Où l'on entrevoit le logiciel «en tant que tel»[24]...

Dans cette même optique de brevet de combinaison logiciel-machine, la proposition de directive de la Commission souhaite en son article 5, que «les états-membres veillent à ce qu'une invention mise en œuvre par ordinateur puisse être revendiquée en tant que produit, c'est-à-dire en tant qu'ordinateur programmé (...)».

---

[12] Vivant [2002].

[13] «un corps déterminé défini par sa composition mécanique ou sa structure chimique» : «un appareil, une machine et plus généralement tout autre objet constituent ainsi un produit.» Lamy Droit Commercial, n°1577.

[14] *ibid.* : «Quant au procédé, il consiste dans la mise en œuvre d'un certain nombre de moyens physiques, chimiques ou mécaniques dont l'agencement permet d'obtenir un certain résultat, consistant soit en un produit, soit en un effet immatériel.»

[15] Consignes d'examen concernant les inventions liées à des programmes d'ordinateur et/ou à des méthodes commerciales, 26/10/00, non publiées.

[16] Décision CRT 1194/97 - 3.5.2, JO OEB, 12/2000, p. 525.

[17] L'expression signifie une revendication de la combinaison matériel-logiciel, d'un support ou d'une machine sur lequel un programme est chargé. Le sens n'est donc pas le même que dans la terminologie employée par un progammeur, où le «produit programme» regrouperait : la documentation (client et technique), le programme (proprement dit : algorithmes et structures de données), et les tests (de vérification et de validation).

[18] Consignes d'examen de l'INPI précitées.

[19] Proposition de directive précitée, article 5, p. 16.

[20] v. la théorie d'origine néerlandaise de la «machine virtuelle» : Le fait de programmer une machine de différentes manières permet de considérer que l'on est en présence de machines de types différents.

[21] Affaire Schlumberger, Cour d'Appel de Paris, 15 juin 1981, PIBD, 1981, III, p. 175.

[22] CRT 3.4.1, 21 mai 1987, T 26/86, «Equipement radiologique» / Koch et Sterzel, PIBD 1988, 432-III, p.185 ; T110/90-3.5.1 - 15 avril 1993 «Document présenté sous une version susceptible d'être éditée»/IBM, JO OEB, 1994, p. 557.

[23] T22/85-3.5.1 - 5 octobre 1988 «Résumé et recherche de documents»/IBM, JO OEB 1990 p.12 ; T38/86-3.5.1, 14 février 1989, «Traitement de texte»/IBM, JO OEB 1990 p. 384.

[24] C'est cette interprétation qu'on pourra discuter afin de remettre en cause la validité du brevet. v. *infra,* partie 2.

Le fait de programmer une machine de différentes manières permet de considérer que l'on est en présence de machines de types différents.

11.1.2 ... D'une invention.

Qu'on le fasse découler du concept d'invention[25] ou de la condition d'application industrielle, l'objet du brevet doit être de caractère «technique». Le procédé revendiqué doit apporter une «contribution technique», qui «se définit comme une contribution à l'état de la technique dans un domaine technique qui n'est pas évidente pour l'homme du métier»[26]. Au-delà de la nécessité d'une distinction entre ce qui est technique et ce qui ne l'est pas, c'est une fonction technique innovante que l'on cherchera à se réserver.

Une innovation.

Deux conditions : une invention nouvelle et non évidente pour «l'homme du métier». à noter que la notion d'originalité (condition d'application du droit d'auteur), objectivisée en «apport intellectuel» pour une adaptation du droit d'auteur au logiciel, est très proche de la nouveauté du droit des brevets.

Nouveauté.

La question du mode de description (v. *supra*) peut ressurgir ici ; adapté, celui-ci permettrait des recherches d'antériorité efficaces, par la détermination d'un état de la technique pertinent.

Dans la pratique des offices (non spécifique au brevet de logiciel), les antériorités citées sont à une écrasante majorité des brevets, et non des logiciels dont le code source a été ouvert ou des articles de revues[27].

Cette condition de nouveauté n'est donc pas insurmontable pour qui veut se faire délivrer un titre : la nouveauté peut être retenue du fait de l'absence d'antériorité de toute pièce, c'est-à-dire d'une modification -même mineure- de «la forme technique ou de la fonction du moyen, du résultat, de la combinaison ou de l'application»[28].

Quoiqu'il en soit, l'argument de l'absence de nouveauté, si elle peut être prouvée, fera annuler le brevet par le juge.

Activité inventive.

Inventivité signifie non-évidence pour l'«homme du métier». L'examinateur de l'OEB doit, par une approche «problème-solution», déterminer le problème technique objectif résolu par un procédé faisant «avancer» l'état de la technique.

La contribution résidera dans les moyens (caractéristiques techniques) revendiqués, dans les effets obtenus, ou dans le problème lui-même : l'appréciation ne peut se faire en connaissant *a priori* le problème, puisqu'il serait intégré fictivement dans l'état de la technique[29] L'important serait de «bien formuler le problème, c'est-à-dire poser la bonne question, et non [de] résoudre ce problème, ce que n'importe quel homme du métier sait faire une fois la question bien formulée»[30]. La CRT de l'OEB le rappelle : «la découverte d'un problème méconnu peut constituer dans certains cas un objet brevetable, même si en elle-même la solution

[25] Défi nition classique : «une solution technique à un problème technique grâce à des moyens techniques et susceptibles de répétition», Mousseron, Traité des brevets, Librairies Techniques, 1984, n° 154, p. 175.

[26] Proposition de directive précitée, p. 14.

[27] Pour une étude statistique sur cette pratique à l'offi ce des brevets américain, v. Greg Aharonian [2000]

[28] Pollaud-Dullian [1999].

[29] «c'est l'énoncé du problème qui n'était pas évident ; et c'est dans cet énoncé que réside l'activité inventive.» Mathély [1991], p. 95.

[30] Smets, précité note 24, p. 44.

revendiquée apparaît rétrospectivement banale et évidente»[31]. Reste que ce problème risque d'être souvent énoncé dans le cahier des charges par le client de celui qui effectuera la demande de brevet.

Le juge, de son côté, recherchera certains critères pour déterminer si l'invention brevetée est réellement inventive : une difficulté surmontée, un préjugé vaincu, la durée de gestation de l'invention, ou encore une rupture décisive avec l'état de la technique antérieur.

Une fonction technique.

Nous ne traiterons ici que de la fonction technique du traitement d'information que réalise le programme, pour lequel l'invention sera toujours une solution technique à un problème technique. Mais l'information traitée, si elle concerne par exemple l'état physique d'un appareil, confèrerait à tout le procédé un caractère technique[32].

Un problème technique.

Comme nous avons pu le voir, «l'important est de bien formuler le problème». Le critère des «considérations techniques» dégagé par l'OEB[33] est la marque d'une érosion des limites du concept de technicité. Il y aura contribution technique (donc délivrance du brevet) si l'on peut relever un «ensemble d'activités faisant intervenir des considérations techniques».

Exemple de problème technique (où ce que représentent les informations traitées est de nature technique) : «la visualisation d'informations sur des événements qui se produisent dans le dispositif d'entrée/sortie d'un système de traitement de texte»[34].

Une solution technique.

La fonction du programme, qui ne doit pas tomber sous le coup d'une autre exclusion de l'article 52 CBE sera de produire un effet «technique».

Tout programme fait produire à un ordinateur des impulsions électriques. Pour être brevetable, l'effet technique doit être «supplémentaire», c'est-à-dire «all[er] au-delà des interactions physique normales»[35] entre programme et ordinateur. «Lorsqu'un programme constitue une modification de données et ne produit aucun effet au-delà du traitement de l'information, il est exclu du domaine de la brevetabilité»[36]. L'effet technique pourrait être produit par une fonctionnalité utilisant les ressources de la machine de manière performante ; mais si seul le logiciel engendre cet effet supplémentaire, le demandeur (ou le breveté) se verra opposer l'exclusion «programme d'ordinateur». Il semble en réalité que l'«effet technique supplémentaire» résultera d'un nouveau périphérique, matériel qui sera breveté en combinaison avec une partie logicielle.

---

[31] T 2/83, 15 mars 1984, JO OEB 1984 p. 265

[32] «La visualisation automatique d'informations sur l'état d'un appareil ou d'un système est essentiellement un problème technique» ; Chambre des Recours, OEB 5 septembre 1988, «méthode de décodage d'expressions et d'obtention de lecture d'événements dans un système de traitement de texte»/IBM, PIBD, 476-III, p.255. «Les codes de commandes d'imprimante doivent être considérés comme des caractéristiques techniques du système de traitement de texte» ; T 110/90, chambre des recours 3.5.1, «document présenté sous une version susceptible d'être éditée / IBM», JO OEB 1994 p. 557. v. aussi (et déjà) la décision Vicom (précitée) dans laquelle la CRT relevait que la méthode revendiquée demeurait abstraite tant «qu'il n'était pas spécifi é que les données représentent une entité physique...».

[33] T769/92-3.5.1, Sohei / «système de gestion universel» - JO OEB 1995 p. 34. La demande portait sur «un système informatique pour différents types de gestion indépendante comprenant, au minimum, des fonctions de gestion fi nancière et de gestion de stock...» dont l'intérêt était de permettre l'utilisation d'un «bordereau de transfert unique» pour l'ensemble des opérations de gestion visées.

[34] T 0115/85, 5 septembre 1988, JO OEB 1990 p. 30.

[35] Décision CRT, 1er juillet 1998, «resynchronisation asynchrone d'une procédure de validation», T 1173/97-3.5.1, JOOEB, 10/1999, p. 620.

[36] CR OEB, Siemens, 12 décembre 1989 T158/88

La Commission nous rassure : «un algorithme[37] défini sans référence à un environnement physique ne présente pas un caractère technique et ne peut donc constituer une invention brevetable.»[38] Il est vrai que d'autres exclusions de l'article 52 CBE concernent les «formules mathématiques», «présentations d'informations» et les «principes scientifiques». L'objet de réservation par brevet pourrait être limité au rapport du principe abstrait à la fonction *précise* revendiquée, c'est-à-dire une implémentation particulière d'un algorithme ; ce serait alors la description d'une algorithmique, science qui étudie l'application des algorithmes à l'informatique, qui serait brevetée.

Le jeu de langage de la doctrine de l'«effet technique» ne permet pas de délimiter précisément et avec certitude le domaine de réservation de ces brevets concernant des programmes d'ordinateur : comment définir ce qui est *technique* ? Chaque activité possède -de manière non exclusive- ses techniques, et on peut en réalité tout mettre derrière ce mot[39]. On pourrait distinguer : les techniques individuelles, les techniques sociales, les techniques intellectuelles, les techniques du réel, pour ne retenir que ces dernières dans la sphère du brevetable : techniques «qui ont pour objet la modification du monde extérieur immédiat, qu'il relève de la nature organique ou inorganique»[40].

Derrière cette exigence de «technique», la question est, plus clairement, de savoir ce qu'on entend par «invention», de l'isolement d'une séquence génétique à la mise en œuvre d'une méthode par ordinateur. Le débat sur la «propriété scientifique»[41] ne ressurgit pas explicitement et pourtant, toute la question est là : que *veut*-on considérer comme invention ?

11.2 La vie du brevet.

La détermination de la sphère de réservation s'effectue par l'examen des spécificités du brevet d'invention en matière de contrefaçon, ainsi que par sa relation avec le droit d'auteur.

11.2.1 La contrefaçon d'un brevet concernant un programme d'ordinateur.

écartons d'emblée le cas particulier de la contrefaçon partielle[42] (dont l'existence est affirmée par la Cour de cassation et majoritairement rejetée en doctrine), qui n'a aucune raison d'exister en cette matière : celui qui a breveté la combinaison A+B+C où A est un programme ne pourra réclamer la protection du seul A, car ce serait rechercher la protection du programme en tant que tel.

Aux actes classiquement interdits (1) viendra s'ajouter une nécessaire adaptation de la théorie des équivalents (2), traditionnelle en droit des brevets.

[37] «Ensemble des règles opératoires propres à un calcul ou à un traitement informatique. -Calcul, enchaînement des actions nécessaires à l'accomplissement d'une tâche.» Petit Robert, 2001.

[38] Proposition de directive précitée, considérant n°13, p. 20.

[39] Weber [1971], p. 63 : « ... on peut parler d'une technique de la prière, d'une technique de l'ascèse, d'une technique de réflexion et de recherche, d'une mnémotechnique, d'une technique pédagogique, d'une technique de la domination politique et hiérocratique, d'une technique de la guerre, d'une technique musicale (d'un virtuose, par exemple), de la technique d'un statuaire ou d'un peintre, d'une technique juridique, etc., toutes étant d'ailleurs susceptibles d'un degré de rationalité extrêmement variable.» Dr. Swen Kiesewetter-Köbinger ajoute, à la suite d'une liste analogue, et non sans humour : «Even with reference the sexual act one speaks occasionally of special techniques.», *On the patent Examination of Programs for Computers*, http://swpat.ffii.org/treffen/2001/bundestag/kiesew/index.de.html

[40] Von Gottl-Ottlilienfeld [1914], cité par Goffi [1996], p. 23.

[41] «Comment admettre qu'on puisse obtenir un titre de privilège pour une simple application industrielle, alors que la découverte de lois scientifiques de premier ordre, qui commandent peut-être toute une série d'applications, ne créerait aucun droit pour le savant ? Celui-ci livre gratuitement l'objet de son travail, cependant le plus méritoire, alors que la société ne semble attacher d'importance qu'au travail terre à terre de l'inventeur, parce qu'il se préoccupe plus immédiatement de la satisfaction des besoins humains.» Roubier [1952],p. 55 et les nombreuses références citées (le débat est ancien). V. aussi de Laat [2000] pp. 187-204.

[42] Qui consiste à protéger une partie isolée d'une invention revendiquée dans son ensemble.

Les actes interdits.

Les prérogatives du breveté.

Dans une logique de revendication de produit, le brevet permet à celui qui en est titulaire d'en interdire la fabrication, l'utilisation et la mise dans le commerce (art. L 613-3 du CPI[43]). La notion est entendue largement : «la mise dans le commerce doit être définie comme toute opération matérielle tendant à mettre un produit en circulation, quelle que soit la qualité dans laquelle l'auteur de cette opération l'a effectuée»[44].

Lorsque c'est un procédé qui est revendiqué, toute utilisation est prohibée sans autorisation du propriétaire du brevet ; et l'interdiction s'étend au produit résultant du procédé.

Mais les droits du breveté souffrent des exceptions : Article L. 613-5 : «Les droits conférés par le brevet ne s'étendent pas :

1. aux actes accomplis dans un cadre privé et à des fins non commerciales ;
2. aux actes accomplis à titre expérimental qui portent sur l'objet de l'invention brevetée.»

Pour faire interdire de tels actes (par le juge), il faudra préalablement que le défendeur n'ait pas réussi à faire annuler le brevet en cause. On peut simplement noter ici qu'étant donné le degré de technicité en cause, le juge dispose en réalité d'«un pouvoir quasi discrétionnaire de décision (qu'en termes optimistes on peut baptiser pouvoir d'équité)»[45]...

L'annulation du brevet par le défendeur.

Le premier moyen d'annulation que le défendeur devra envisager est celui du défaut de nouveauté et d'activité inventive (v. *supra*).

Comme nous l'avons vu, l'OEB interprète les revendications dans leur ensemble pour déterminer si leur objet est «technique». Il nous semble que le défendeur pourra chercher à prouver que le seul aspect innovant de l'ensemble est un programme d'ordinateur ; et que, par conséquent, le breveté, demandeur à l'instance, recherche la protection d'un programme en tant que tel.

Les actes interdits ne concernent pas uniquement l'utilisation du procédé ou produit *stricto sensu*, un équivalent sera aussi considéré comme contrefaisant.

L'indéfinissable «moyen logiciel» équivalent.

La définition d'une fonction technique équivalente.

La théorie des équivalents permet d'interdire l'utilisation de moyens *équivalents* à ceux qui sont brevetés, c'est-à-dire un moyen de forme différente mais qui remplit la même fonction en vue de parvenir à un résultat identique. La fonction, en droit des brevets, se définit comme l'ensemble des «effets premiers» réalisés par l'exécution du procédé ou du produit.

L'équivalent sera alors ce qui permettra d'obtenir le même effet technique grâce à un moyen de forme, de structure différentes. à l'évidence, le domaine revendiqué peut être très large[46].

[43]«Sont interdites, à défaut de consentement du propriétaire du brevet :

1. la fabrication, l'offre, la mise dans le commerce, l'utilisation ou bien l'importation ou la détention aux fi ns précitées du produit objet du brevet ;
2. l'utilisation d'un procédé objet du brevet ou, lorsque le tiers sait ou lorsque les circonstances rendent évident que l'utilisation du procédé est interdite sans le consentement du propriétaire du brevet, l'offre de son utilisation sur le territoire français ;
3. l'offre, la mise dans le commerce ou l'utilisation ou bien l'importation ou la détention aux fi ns précitées du produit obtenu directement par le procédé objet du brevet.»

[44]Paris, 3 déc. 1985 : PIBD 1986, 388, III, 130

[45]Foyer et Vivant [1991], p. 168

[46]British Telecom tente ainsi de faire appliquer son brevet portant -prétendument- sur les liens hypertextes en interprétant le texte

Brevet de résultat.
Le moyen serait une méthode, la description, l'implémentation de l'algorithme. Déjà «la distinction du résultat et de la fonction technique du moyen [était] hérissée de difficultés»[47] lorsqu'on traitait d'inventions mécaniques ; protéger par brevet la fonction technique d'un programme d'ordinateur reviendrait à permettre la réservation du résultat lui-même. Et breveter un résultat est traditionnellement prohibé en droit français.

Dans le domaine du logiciel, on pourra s'en remettre à la sagesse du juge, qui déboutera le breveté arguant de l'utilisation d'un moyen équivalent au sien. Mais le choix de rejeter en bloc un pan entier de la doctrine classique du droit des brevets nous apparaît devoir peser sur le législateur plutôt que sur le juge.

Ainsi, on peut arriver à une situation quelque peu absurde où, lorsque le brevet est accordé pour une invention constituée par l'ensemble matériel et logiciel, « substituer un logiciel nouveau au logiciel qui aurait permis d'obtenir le brevet pourrait être considéré comme une contrefaçon par équivalent, fournir un tel logiciel (le commercialiser et l'offrir au public) pourrait devenir un acte de contrefaçon par fourniture de moyen»[48]. Le logiciel faisant fonctionner un matériel donné (on songe au pilote de périphérique) ne pourra être reproduit pour que ce matériel fonctionne, par exemple, pour un autre système informatique. Ce qui n'est pas sans difficultés lorsqu'on étudie les règles relatives au droit d'auteur, notamment la garantie d'une possible interopérabilité entre systèmes informatiques.

11.2.2 Vers un cumul de protection.

Le brevet protègerait les principes sous-jacent d'un programme, dont l'écriture -la mise en forme- est protégée par le droit d'auteur. La mise en œuvre du principe est loin d'être aisée. La coexistence, c'est-à-dire la définition de deux objets de droit différents, apparaît impossible à mettre en œuvre en pratique. C'est pourquoi le système de cumul de protection est le seul envisageable[49].

Droit d'auteur et protection de l'idée.

Côté droit d'auteur, le logiciel est assimilé à une œuvre littéraire. C'est la forme qui est protégée, ce qui inclut la structure de l'œuvre.

Une protection de la seule forme.
Le principe du droit d'auteur est de ne protéger que la forme d'expression d'une œuvre, et non l'idée sous-jacente. Concernant le logiciel, la protection concerne évidemment le code source, mais aussi le code objet, bien que sa forme soit volatile et dépende notamment de l'architecture matérielle sur laquelle le source est compilé.

Bien sûr cette forme doit être originale, c'est-à-dire que le programmeur doit avoir eu des choix dans l'écriture du programme : avoir fourni un «effort intellectuel personnalisé». Les critères retenus par les juges pour caractériser l'originalité d'un logiciel peuvent se rapprocher de ceux utilisés pour évaluer l'activité inventive d'une invention (v. *supra*).

Une protection de la structure.
L'exclusion des idées de la protection par la propriété littéraire et artistique est de principe, mais -une fois

de ce brevet par le biais des équivalents : «blocks of information» serait tout fi chier html, une «complete adress» représenterait toute URL internet.

[47] Roubier [1954], p. 82.

[48] Vivant et al. [2001], p. 109.

[49] «La protection peut être cumulative au sens où un acte impliquant l'exploitation d'un programme particulier peut porter atteinte aux droits d'auteur protégeant le code du programme et à un brevet protégeant les idées et principes sous-jacents à ce programme.», proposition de directive précitée, p. 9.

de plus- la limite est difficile à tracer : «Il est impossible de fixer une limite précise à laquelle s’arrête la contrefaçon punissable, à laquelle commence le plagiat toléré.»[50]

Le critère de la composition, formulé par Desbois, permet de protéger «le développement de l’idée générale»[51] par exemple le plan, l’ordonnancement de l’ouvrage ; en appliquant ce critère au logiciel on songe à la structure interne, l’architecture du programme.

Une articulation complexe.

Le seul exemple de cumul de protection entre un titre industriel et le droit d’auteur est celui des dessins et modèles industriels. Et le «caractère inséparable ou séparable de la création de forme et de la création technique est une opération fort délicate» en cette matière[52]. *a fortiori*, dans le domaine du logiciel, la distinction sera extrêmement épineuse.

Des antagonismes entre droit d’auteur et brevet.

Il convient ici de souligner que le brevet laisse une plus grande liberté aux tiers que le droit d’auteur, qui ne permet qu’une reproduction à titre de sauvegarde (et non simplement dans un «cadre privé») et enferme la décompilation («à titre expérimental») dans de strictes conditions (l’objectif d’interopérabilité entre systèmes).

De plus, la description d’un algorithme dans une demande de brevet pourrait reprendre la même structure d’écriture que le programme lui-même (un algorithme est une structure). Dans le cas d’un programme écrit avec un langage informatique faiblement typé, où une adéquation structure-écriture existe, retenir la protection de la structure par le droit d’auteur reviendrait à reconnaître que la publication du texte du brevet serait une contrefaçon, voire même la protection de l’algorithme par le droit d’auteur !

Un cumul offrant plus de liberté ?

Des dispositions légales particulières sont évidemment nécessaires pour organiser ce cumul, du type : «les dispositions concernant le droit d’auteur ne portent pas atteinte à celles des brevets, et inversement»

La première partie de cette proposition est énoncée par la Commission : «l’exercice d’un brevet s’appliquant à une invention mise en œuvre par ordinateur ne doit pas interférer avec les exceptions accordées en vertu de la législation sur le droit d’auteur aux concepteurs de logiciels par les dispositions de la directive 91/250/CEE.»[53] Autrement dit, la décompilation sera toujours possible.

Et inversement ? Il faudrait ajouter : les dispositions du droit d’auteur ne portent pas non plus atteinte aux exceptions (particulièrement l’exception pour usage privé à des fins non commerciales) du droit des brevets. Mais ce serait rendre sans effet la législation du droit d’auteur spécifique au logiciel : la limitation à la copie de sauvegarde (plus restrictive que la copie privée «classique») serait alors sans objet pour tout programme breveté !

Conclusion.

En l’état actuel des textes, les incohérences du système juridique de protection des logiciels sont flagrantes, tant du point de vue des principes que de leur mise en œuvre. A l’évidence, le droit des brevets a besoin de profondes adaptations si l’on veut faire entrer le programme d’ordinateur dans la sphère de réservation de la propriété industrielle. Dans l’hypothèse d’une réelle volonté politique (que le domaine juridique ne fait

[50] Pouillet, cité par Lucas, Traité de Droit de la Propriété Intellectuelle, p. 240.
[51] Cité par Lucas, *Ibid.*
[52] Bertrand-Doulat [1997], p. 63
[53] Proposition de directive précitée, p .9.

finalement que mettre en forme) de mise en œuvre de ces adaptations, il nous semble que le «brevet concernant un programme d'ordinateur» ne sera plus véritablement du brevet tel que nous pouvons l'entendre aujourd'hui (tout comme le droit d'auteur spécifique au logiciel se rapproche plus en réalité d'un droit *sui generis*), mais un titre industriel particulier.

Conclusion Générale. Vers une Nouvelle Économie du Logiciel ?

Jean-Michel Dalle,
Université de Paris 6 & IMRI-Université de Paris Dauphine, Jean-Michel.Dalle@admp6.jussieu.fr.

12.1 Introduction : À propos de quelques arbres qui auraient pu cacher la forêt.

Qu'en est-il vraiment de la «nouvelle économie du logiciel» ? C'est ainsi que nous avons désigné et exprimé[1] la sensation partagée par la grande majorité des acteurs de l'informatique que des mutations en cours allaient à nouveau révolutionner l'industrie du logiciel. Cette sensation faisait essentiellement suite à l'apparition des logiciels libres et ouverts au cours de la décennie 90, dont les positions commençaient à s'affirmer en termes de modèles économiques et de parts de marché, et même à inquiéter leurs concurrents propriétaires, et dont l'émergence relançait aussi l'espoir plus profond qu'apparaisse «enfin» dans le monde de l'informatique un modèle viable de production à partir de composants logiciels ou COTS, pour «components-off-the-shelf».

Qu'en est-il réellement ? Force est tout d'abord de constater que le contexte économique autour de l'an 2000 a eu son importance, et que la période récente a été pour le moins troublée par les hauts et les bas de la «nouvelle économie», par la formation puis l'explosion de la bulle financière associé aux dot-coms et généralement aux start-ups. Quelle importance ce contexte aura-t-il eu sur les logiciels libres : pour le dire en un mot, y a-t-il eu «path-dependence» (David [1985]) ? Bref, quelle est la part de cette histoire récente dans la situation actuelle et le devenir de la nouvelle économie du logiciel ? Cette question est largement insoluble, au moins à ce stade, mais il est clair que la nature des financements disponibles oriente les choix de modèles économiques des entreprises, puisqu'une dimension essentielle d'un modèle économique est l'investissement auquel il conduit, et sur combien de temps. A un moment donné, des modèles économiques jouant sur la prise très rapide de parts de marché et donc sur des stratégies de standardisation - devenir le standard dominant -, et associés donc à des investissements considérables pendant de longues périodes, ne faisaient pas peur aux investisseurs. C'est ce qui aura notamment permis l'essor initial des «distributeurs», essentiellement de Linux (Red Hat, MandrakeSoft, SuSE, Caldera, etc.) À l'évidence, ces investissements correspondaient à des conditions tout à fait particulières sur les marchés financiers, qui n'ont plus cours aujourd'hui, et dont il n'est pas certain qu'elles se représentent fréquemment. Pour autant, le cas des distributeurs de Linux est un cas particulier, qui se caractérise avant tout par des stratégies de standardisation, et qui n'a même pas vraiment d'équivalent pour d'autres logiciels libres. Il se peut donc qu'il ait joué le rôle de l'arbre et caché la forêt de toutes les autres sociétés associées au développement des logiciels libres, et plus généralement des évolutions qui caractérisent la nouvelle économie du logiciel, que cette conclusion va tenter de mettre en lumière. Certes, ces sociétés ont dans une certaine mesure bénéficié également d'un contexte économique favorable sur les marchés financiers au travers d'investissements en fonds privés et notamment en fonds de capital-risque. Mais on peut a contrario interpréter également ce phénomène comme ayant permis une exploration plus rapide qu'à l'accoutumée, de la question des modèles économiques notamment, c'est-à-dire comme ayant permis de progresser plus vite sur la courbe collective d'apprentissage[2]. Il s'agit d'une vision classiquement schumpétérienne, mais dont

[1] Au moment du lancement de ce projet, qui avait pour principaux objectifs de favoriser l'apprentissage collectif autour de ces questions nouvelles et complexes, et d'aider une communauté de réflexion et de recherche à se rassembler, notamment au travers de l'organisation de workshops qui ont permis à des acteurs praticiens et académiques de se rencontrer et d'échanger, mais aussi grâce aux travaux de jeunes chercheurs intéressés par ces questions. Merci au demeurant à Nicolas Jullien d'avoir bien voulu prendre le relais de la coordination de ce projet durant ses derniers mois, alors que j'avais été contraint de me consacrer à d'autres tâches. Le rapport final lui est largement dû. Merci plus généralement à Nicolas Jullien et à Mélanie Clément-Fontaine, sans lesquels ce projet n'aurait sans doute jamais vu le jour. Merci aussi à Laurent Kott, dont le soutien et l'intérêt pour ces questions à joué un rôle très important pour nous faire avancer, ainsi qu'à Marie Coris, Yves Rougy, François Horn, Franck Macrez, Laure Muselli et Aymeric Poulain-Maubant pour leurs excellentes, et pertinentes, contributions. Merci enfin à tous les participants et intervenants des deux workshops organisés à l'INRIA, et en particulier à Christian Genthon, Michel Bidoit, Gérard Roucairol et Hughes Rougier. D'un point de vue plus personnel, d'innombrables discussions m'ont aidé à mieux comprendre l'économie des logiciels libres, et je serais bien en peine de citer tout ceux qui ont ainsi contribué, soit en apportant de nouvelles idées/fonctionnalités, soit en débuggant certaines de mes idées. Parmi eux, je pense toutefois particulièrement à mes nombreuses discussions avec Michel Bidoit, à nouveau, Gérard Giraudon, Hugues Rougier, notamment lors de la rédaction du rapport du groupe logiciel du Commissariat au Plan (Commissariat au Plan, 2002) et Guillaume Rousseau, à peu près une fois par semaine. Quoi qu'il en soit, il reste des bugs - ils me sont dus - et surtout de nouvelles idées à apporter : toutes les contributions sont donc évidemment les bienvenues !

[2] C'est aussi avant tout en termes de courbe d'apprentissage, et d'apprentissage collectif autour de la question des logiciels ouverts,

l'originalité est d'être appliquée alors à une rupture plutôt économique que technologique : de la même manière que toute rupture technologique est accompagnée de son lot d'essais et d'erreurs, incarnées souvent par autant de sociétés, une rupture socio-économique s'accompagne ici également d'une telle phase d'exploration. Dès lors, cette exploration est d'autant renforcée que le nombre de nouvelles entreprises créées est grand et qu'elles explorent substantiellement de nombreuses voies différentes[3]. Cette dynamique schumpétérienne s'applique-t-elle vraiment lorsqu'il n'est plus question de promouvoir une technologie véritablement nouvelle et révolutionnaire ? Notre conjecture serait ici que cette dynamique est toujours valable mais moins forte, puisque les dynamiques de marché ne sont pas la même : il ne s'agit pas de vendre un produit nouveau, que les concurrents n'auraient pas. La capacité à contester des marchés existants, et à se protéger soi-même par des barrières à l'entrée suffisantes, n'est pas la même. En d'autres termes, la rupture aurait plus de chances de *ne pas* s'imposer - de rater l'occasion de s'imposer. Pour des technologies caractérisées par des effets de réseau, la question est alors aussi une question d'économie de la standardisation, dont les énoncés sont d'une nature différente par rapport aux questions de rupture technologique, et méritent d'être étudiés en tant que tels (voir e.g. Foray [2002] pour une revue récente de l'économie de la standardisation). Contrairement à d'autres des nouvelles technologies apparues ces derniers années, les logiciels libres ne disposent pas forcément des avantages comparatifs que confère une percée technologique majeure. Il s'agit plus pour eux de stratégies de pénétration de marché ou de remplacement de produits existants, et donc de standardisation progressive en présence de forts effets de réseau. Dans le domaine des distributeurs de Linux, il serait à nouveau tout a fait erroné de raisonner en termes de modèles économiques pour les entreprises concernées, alors qu'il s'agit avant tout d'une question de standardisation. Une stratégie de standardisation possible lorsque les financements sont facilement accessibles consiste à tenter de gagner le plus vite possible, si possible, des parts de marché. Cette stratégie s'est avéré assez peu fructueuse, et est désormais caduque : d'autres stratégies sont apparues en la matière, qui toutes ont à voir avec des stratégies de standardisation, qu'il s'agisse de l'introduction en bourse de Red Hat au NASDAQ et dans une moindre mesure de MandrakeSoft au Marché Libre de la Bourse de Paris, avec la visibilité qu'elle procure au moins autant que les financements, du regroupement de plusieurs distributeurs sous la bannière d'United Linux ou encore de la décision de SuSE de proposer une interface entre sa distribution et le pack Office de Microsoft. De même, certains distributeurs comme MandrakeSoft ont pu s'orienter vers des stratégies de niches, comme par exemple l'éducation, et tendent alors à différencier leurs produits. Mais la différence est alors très mince avec des éditeurs traditionnels : seul l'espace des stratégies de standardisation possibles est peut-être légèrement différent.

Ainsi, dans cette conclusion, il s'agit plutôt pour nous de mettre en évidence les phénomènes qui caractérisent réellement la nouvelle économie du logiciel, et ce qu'elle est susceptible d'apporter de «neuf». Ainsi, nous allons dans un premier temps insister sur le fait que l'économie associée aux logiciels libres est avant tout une économie de bundles et plus précisément de couches (layers). En un sens, toute l'économie du logiciel est une économie de bundles, mais la gratuité des logiciels libres rend ce phénomène plus particulièrement

---

qu'il faut interpréter les contributions présentées dans ce rapport, et plus généralement le projet RNTL NME/NEL. Cela conduit, et aussi l'état d'esprit que nous décrivions plus haut (cf. supra, note 1), à faire preuve d'une certaine prudence, et à éviter les réponses «défi nitives» - et presque instantanément obsolètes. Ainsi, nous nous sommes principalement attachés dans le cadre de ce projet à un travail d'éclaireurs et d'éclairage qui nous semblait prioritaire. C'était aussi faute de données suffi samment accessibles au moment où ce projet a été lancé : les premiers chiffres en matière de coût des logiciels libres (TCO) sont ainsi récents, et encore relativement contradictoires puisque nous n'avons en particulier connaissance à ce jour d'aucune étude qui prendrait réellement en compte les coûts de changements et d'apprentissage, et non pas simplement les coûts d'acquisition purs et simples. Plus important encore, l'ampleur de l'adoption des logiciels libres, au delà de quelques chiffres connus - et certes parlants - en termes de parts de marché, est diffi cile à évaluer puisqu'elle concerne d'abord ce qui se passe à l'intérieur des entreprises, alors même que cette partie immergée de l'iceberg commence seulement à vouloir apparaître aujourd'hui : il y a encore peu, il n'était pas si facile de savoir qui utilisait Linux dans telle ou telle entreprise, voire dans telle ou telle administration, voire encore qui dans telle ou telle SSII «traditionnelle» fournissait des services autour de logiciels libres. Il nous semble aujourd'hui que si ce projet devait avoir une suite, elle devrait aller en ce sens, et contribuer à éclairer maintenant ces questions de manière véritablement quantitatives, au moment même où ce marché commence véritablement à frémir.

[3] Tout simplement, une telle vision schumpétérienne animait aussi un certain nombre d'acteurs et d'observateurs de la «nouvelle économie».

significatif encore. C'est typiquement ainsi que l'on peut facilement interpréter l'apparition des sociétés de service en logiciels libres (SS2L) et des promoteurs, mais aussi, de manière plus récente, l'apparition de ceux que nous appellerons «intégrateurs spécialisés» et «quasi-éditeurs». Ce dernier modèle est particulièrement lié à la nature fondamentale de la nouvelle économie du logiciel, qui est de promouvoir un modèle de composants - une véritable «révolution modulaire», pour reprendre le mot de Langlois [2001]. Plus exactement, il s'agit d'une économie où les intégrateurs de systèmes Pavitt [2002] sont appelés à jouer un rôle important, et où l'ouverture des sources assure la présence sur étagère de composants logiciels facilement adaptables.

12.2 Une économie de «bundles».

L'économie associée aux logiciels libres est avant tout une économie de «bundles». Puisque que l'on ne peut pas vendre directement les logiciels libres eux-mêmes - sans qu'il y ait de règle, ils sont presque toujours gratuits - l'objectif général est de vendre quelque chose d'autre avec lequel ils sont «associés», et qui idéalement leur est nécessaire, et au moins leur est utile. Un cas typique de bundle en économie est celui des rasoirs à lame : on pourrait presque donner gratuitement le rasoir, du moment que l'on vende ensuite les lames. Dans le domaine des logiciels libres, cette situation est simplement poussée à l'extrême puisque le «rasoir» est vraiment donné, voire même accessible, gratuitement, mais la logique demeure tout à fait la même. Ainsi, le packaging, la documentation, l'aide en ligne, et d'autres services similaires peuvent être utiles pour les usagers des logiciels libres, et sont donc susceptibles d'être vendus en «bundle» avec ces mêmes logiciels. Le packaging et l'aide en ligne sont toutefois relativement peu utiles dès lors que l'on s'adresse à une population de développeurs aguerris : il s'agit vraisemblablement d'une autre raison qui explique les difficultés qu'ont connues certains purs «distributeurs» de logiciels libres. À l'autre extrême, une société comme O'Reilly qui s'est spécialisée dans l'édition de livres et de manuels dédiés aux logiciels libres, a connu un succès retentissant : il s'agit pourtant également d'un modèle de bundle, ici tourné vers la fourniture de la documentation. Dans tous les cas, c'est donc la nature de la demande qui conditionne la nature des bundles susceptibles d'être proposés, sur des marchés éventuellement segmentés.

De manière générale, l'association de deux biens peut produire une valeur ajoutée par rapport à la valeur de chacun pris isolément, et cette valeur ajoutée peut être partiellement appropriée par des acteurs économiques. Dans le cas des logiciels, l'un des bundles les plus naturels sur les marchés professionnels consiste à associer logiciels et services. L'installation d'un logiciel n'est en effet pas chose aisée si les compétences n'existent pas chez le client, plus même, une adaptation à ses besoins particuliers est souvent nécessaire et presque toujours utile, ceci alors que la nature même des biens logiciels permet justement leur adaptation de manière assez aisée. C'est ce simple raisonnement qui explique le largement succès des sociétés de service dans le monde de l'informatique : c'est aussi le cas pour les logiciels libres, avec les «SS2L» (Sociétés de Service en Logiciels Libres), qui font tout simplement l'analogue des SSII. Dans le cas des logiciels libres, la problématique de recrutement est toutefois particulièrement essentielle car elle concerne des développeurs généralement indépendants et issus des communautés de développement associées aux logiciels libres. Ainsi (cf. l'article de Marie Coris dans ce rapport), les SS2L font généralement des efforts vis à vis des communautés de développeurs, en accordant du temps à leurs développeurs pour contribuer à des projets de logiciels libres, en initiant de tels projets, en embauchant des développeurs connus et en les laissant libres de développer ce que bon leur semble, en respectant la culture spécifique des programmeurs issus des communautés, voire même en adoptant des modes organisationnels spécifiques, qui avoisinent des formes coopératives, afin de faciliter leur recrutement, et afin de développer une culture d'entreprise qui leur permette de fidéliser leurs employés et d'en attirer de nouveaux. Ces pratiques sont donc aussi à interpréter en termes de signalling vis-à-vis du marché du travail des développeurs spécialisés dans les logiciels libres.

Le modèle économique des SS2L est l'un des plus fréquemment cités lorsque qu'il est question de logiciels libres et de nouvelle économie du logiciel. Il est parfois même considéré comme le seul modèle économique viable dans ce domaine. Cette nouvelle vérité, qui a succédé au foisonnement de tous les modèles économiques

explorés en liaison avec des logiciels libres durant la période de la «nouvelle économie», mérite tout autant d'être questionnée. Comme on l'a vu, des sociétés comme O'Reilly, mais aussi certains éditeurs de jeux en réseau qui offrent le logiciel et ses sources, mais facturent le service d'abonnement pour participer au jeu lui-même, ce qui leur permet non seulement d'attirer des joueurs mais aussi de bénéficier de leurs contributions à son développement, continuent à explorer, certaines avec un certain succès, des voies originales[4].

Mais le panorama ne s'arrête pas là, et ces modèles de bundles sont en fait avant tout, dans le cas des logiciels libres, des modèles de «couches[5]». Le principal impact des logiciels libres se situent au niveau des couches dites «basses», qui touchent ainsi au systèmes d'exploitation et à généralement ce que l'on est convenu d'appeler «middleware», qui peut aussi aller des serveurs d'applications aux outils de développement, en passant par les langages (cf. Figure 1). Il y a une raison fondamentale à cela, qui tient à la production même des logiciels libres : ils sont avant tout produits par des communautés de développeurs pour répondre à leurs besoins. Ces besoins se situent plus au niveau des outils et de la configuration des machines et en particulier des serveurs, c'est-à-dire précisément aux niveaux évoqués. Il est clair que des logiciels libres existent dans d'autres domaines, mais il existe dans le middleware une adéquation particulièrement forte entre les motivations des développeurs et la nature des outils produits, qui sont utiles aux développeurs et s'adressent avant tout à eux. Dès lors, lorsque le middleware est libre, il peut être associé en bundle soit avec des services voire le cas échéant, et lorsque cela est juridiquement possible, avec des solutions logicielles propriétaires (bundle «vers le haut»), soit avec du matériel ou hardware (bundle «vers le bas»)[6].

***Figure 12.1* —** Les différentes «couches» informatiques.

Paramétrages, Logiciels Spécifiques
Progiciels
« Middleware » Langages Syst. d'exploitation ...
« Hardware » Microprocesseurs Circuits « logiciels » ...

Dans ce dernier cas, les acteurs concernés sont alors les constructeurs, et les principaux - IBM mais aussi HP, Sun ou Bull en France - sont précisément devenus des «sponsors» ou encore des «promoteurs» des logiciels libres. Il ne s'agit pas ici d'un nouveau modèle économique à proprement parler, mais plutôt d'une extension de leurs propres modèles économiques, dans le cadre également de leurs relations de dépendance ou d'indépendance avec les grands éditeurs de logiciels. Le rôle de ces promoteurs, qui deviennent de plus

[4]OSDN, le service en ligne propriété de VA Software (ex-VA Linux), et qui inclut notamment slashdot, est un autre exemple de ce type.

[5]Merci à Hughes Rougier et à François Horn pour m'avoir aidé à y voir plus clair sur ces questions.

[6]Au sein du middleware, il faut également tenir compte de quelques bundles entre des logiciels libres des solutions logicielles propriétaires, essentiellement lorsque ces derniers sont en fait des outils de développement : c'est par exemple cas des plates-formes de développement collaboratif éditées en mode propriétaire par Collab.Net et maintenant par VA Linux, depuis sa transformation en VA Software. Le développement de Linux utilise lui-même un autre logiciel propriétaire similaire, BitKeeper. Quelques bundles vers le haut similaires sont apparus entre des solutions libres et des solutions propriétaires - nous avons cité plus haut l'exemple de SuSE et de Microsoft Offi ce - mais ces tentatives restent l'exception plutôt que la règle.

en plus les «patrons», au sens historique et anglo-saxon de ce terme, c'est-à-dire les financeurs, des communautés et des logiciels libres, mériterait sans doute d'être plus complètement éclairé que nous n'avons pu le faire dans le cadre de ce projet. S'il s'agit ici encore de bundles, le rapport, et peut-être à nouveau le rapport de force, qu'ils entretiennent avec les logiciels libres et les communautés qui les développent est en effet potentiellement ambigu du fait notamment de leur taille et de leurs positions sur leurs marchés. L'une des principales questions tient à l'ampleur des investissements réalisés et au fait qu'ils puissent être dédiés à telle ou telle plate-forme matérielle : en d'autres termes, cela revient à rendre un logiciel peu ou prou propriétaire, même s'il est libre ou ouvert, ou à tendre à imposer un standard de fait du côté du matériel. De ce point de vue, une manière assez générale d'interpréter les relations entre les sociétés dédiées aux logiciels libres et les communautés de développeurs - qui ne s'applique pas qu'aux constructeurs - consiste à considérer que ces relations de collaboration où existe un risque d'opportunisme (défection de l'entreprise mais aussi défection de la communauté qui orienterait par exemple ses développement dans une direction néfaste pour l'entreprise) sont généralement réglées par des échanges mutuels d'«otages» (Williamson [1996]). Ces otages sont typiquement d'un côté les développeurs embauchés par ces entreprises, et de l'autre côté les effets de réputation qui peuvent en effet être facilement ternis par la communauté, dès lors que ces entreprises adopteraient des stratégies qui les éloigneraient du Libre[7]. Mais ce raisonnement ne vaut pas pour des multinationales dont la taille et les parts de marché les protègent et les rendent quasi invulnérables. Il n'y a plus alors d'échanges d'otages qui tienne. Pour ces entreprises, le développement de produits et de services dédiés à des logiciels libres correspond au fait de saisir une opportunité de marché, et il est tout à fait évident que ces opportunités seront exploitées dans le sens du profit, ce qui impliquera souvent, lorsque ces entreprises seront aussi des constructeurs d'ordinateurs, une spécification plus poussée aux besoins de ses plates-formes matérielles : pour des raisons d'efficacité mais aussi, évidemment, pour accentuer un avantage comparatif. Les risques d'opportunisme de la part de ces sociétés existent donc, et ne sont pas vraiment réglés par l'existence d'otages contrairement à ce qui peut se passer pour des sociétés de plus petite taille.

12.3 Le rôle accru des (nouveaux) intégrateurs.

Quoi qu'il en soit, le principale rôle que jouent SS2L comme promoteurs est celui d'intégrateur. Cela signifie qu'elle proposent des solutions sur mesure à leurs clients[8], qui incluent selon les cas un bundle complexe de services (de développement, de maintenance, d'infogérance, etc.), d'éléments logiciels (développement spécifiques, progiciels, solutions open-source, etc.), de middleware et de hardware. Ces bundles complexes de biens et de services correspondent précisément aux besoins justement complexes des clients, notamment professionnels. Dans ce contexte, l'apparition des logiciels libres correspond à l'émergence de «nouveaux intégrateurs», qui peuvent intégrer sans être contraint de maintenir des relations privilégiées avec tel ou tel éditeur, et à un développement supplémentaire modèle de l'intégration. Les grandes SSII non liées à des constructeurs n'ont de ce point de vue pas encore véritablement pris parti dans cette bataille des intégrateurs qui s'annonce, sans doute d'abord du fait de leurs liens privilégiés avec les grands éditeurs ; pour autant, il n'est pas impossible que leur intérêt puisse les rapprocher des logiciels open-source d'une manière presque plus indépendante que celle des constructeurs. C'est aussi dans le contexte de ces prochaines batailles entre intégrateurs pour le contrôle des clients que l'on peut interpréter l'évolution de nombreuses sociétés liées aux logiciels libres vers ce que l'on pourrait appeler un modèle économique d'«intégrateur spécialisé», aussi paradoxale cette expression puisse-t-elle paraître au premier abord. Il peut s'agir de distributeurs comme Red Hat qui sont ainsi tout autant des distributeurs que des SS2L, mais plus spécifiquement tournées alors vers tel ou tel (éventuellement un petit nombre) de logiciels libres. Il peut s'agir de SS2L qui se spécialisent, sur un logiciel comme Sendmail ou Zope par exemple, ou encore de constructeurs-intégrateurs qui comme IBM

[7] En ce sens, OSDN est, à nouveau, le principal otage ou plutôt «gage» que VA Software peut encore offrir aux communautés de développeurs, dans la mesure où SourceForge est maintenant à peu près devenu un logiciel propriétaire.

[8] Cf. les annonces d'IBM fi n 2002 sur le développement de services «on demand».

associent leur marque à Linux. Dans tous les cas, il s'agit toujours de modèles économiques basés sur des bundles, mais affinés, ou spécialisation et image de marque joue un rôle plus important dans la compétition qui est alors un marché de niche dans la plupart des cas, sauf peut-être pour IBM et Red Hat, c'est-à-dire pour les intégrateurs «spécialisés Linux». Ces intégrateurs spécialisés offrent à leurs clients des services «autour» d'un ou de quelques logiciels, mais qui sont en réalité souvent plus larges. Simplement, le niveau de support offert pour d'autres logiciels n'atteint pas le même niveau, et on atteint ici l'idée d'une différenciation par la qualité des services (Jullien [2001]), et à nouveau d'un signalling adapté, auprès de clients autrement attirés par des intégrateurs plus «généralistes» et susceptibles de leur proposer une solution complète - intégrée - à leur problème complexe. Il s'agit donc aussi d'une tentative pour segmenter plus ou moins le marché.

Quoi qu'il en soit, la période qui s'annonce est celle des intégrateurs, spécialisés ou non. Ce qui n'empêche pas bien entendu coexistence et compétition entre solutions logicielles libres et propriétaires (Dalle et Jullien [2003]).

Et le futur de l'informatique se rapproche alors en un sens du présent de l'automobile, dans un monde dont les mots-clefs sont intégrateurs («systémiers») mais aussi, sur l'autre face, «modularité» puisque les intégrateurs font en sorte d'assembler des modules afin de fournir des solutions sur mesure à leurs clients. Cette évolution s'interprète plus généralement comme une véritable rupture avec la «grande entreprise» traditionnelle (Pavitt [2002]), dans un contexte où la fabrication stricto sensu tend à se déplacer vers des pays où la main d'œuvre est bon marché : la valeur ajoutée, dans des économie développées, est alors surtout liée à la conception et au développement de produits de plus en plus sophistiqués. Ainsi, les grands constructeurs automobiles sont devenus avant tout des systémiers, qui maintiennent en interne quelques actifs spécifiques clefs, et qui font appel à des sous-traitants pour tous les autres modules. Et, comme on vient de le voir, il en va de plus en plus de même pour les intégrateurs informatiques, et particulièrement pour les «nouveaux intégrateurs» qui bénéficient de l'apparition des logiciels libres : ces ensembliers informatiques rendent à leurs clients des services sur mesure, avec toutes les «options» dont ils souhaitent disposer, bien au delà de la couleur de leur véhicule. Ces services sont rendus en associant et en intégrant des modules dont certains seront matériels et d'autres logiciels, dont une partie libre.

12.4 La deuxième révolution modulaire de l'informatique.

Il s'agit d'un point primordial, puisque ces évolutions s'accompagnent d'une «révolution modulaire» (Langlois [2001]) dans le monde du logiciel, la deuxième en un sens pour l'informatique après celle des composants matériels, qui met l'accent sur la fourniture de services, à la conception de produits nouveaux et variés grâce à l'assemblage de composants. Cette révolution modulaire est avant tout permise par l'existence de composants logiciels dont les sources sont ouvertes : seule cette ouverture permet en effet l'adaptation de ces composants - précisément facile puisqu'il s'agit de logiciel et de code - à chaque besoin ou situation particulière, qu'il s'agisse d'une plate-forme matérielle particulière ou du besoin particulier d'un client. Les logiciels fermés, quel que soit le degré de codification et de spécification des interfaces, ne permet pas vraiment ce mode de développement et de «fabrication» des solutions logicielles et/ou matérielles, et les dynamiques de marché qui tendent à rendre même les interfaces propriétaires s'oppose encore plus à cette évolution[9]. À cette possibilité de rendre des composants open-source facilement inter-opérables s'ajoute aussi leur fiabilité, ainsi que la possibilité de les maintenir durablement puisque leurs sources sont très largement accessibles, sans parler du support qu'apporte l'existence d'une communauté de développeurs, qui apparaît dans de nombreux cas assez largement pérenne, et de sociétés spécialisées.

Ces sociétés peuvent évidemment être des intégrateurs spécialisés, qui agiraient alors en sous-traitants

[9] C'est aussi en ce sens - afi n de lever cette hypothèque - que l'on peut typiquement interpréter l'apparition de modèles de «partage» des sources entre des éditeurs propriétaires et des intégrateurs, et qui vont dans le sens d'une économie modulaire où les solutions seraient aussi des composants.

d'intégrateurs plus généralistes, et représenteraient ainsi une véritable interface entre ces derniers et les communautés, considérés alors comme des marchés dédiés et quasi-internes du travail. Mais c'est aussi dans ce contexte qu'il faut interpréter l'apparition de «quasi-éditeurs» de modules open-source, dont MySQL est un exemple, et qui utilisent des schémas de «doubles licences» afin de distribuer leurs produits à la fois sous la forme d'un logiciel libre, pour des applications non commerciales, et sous la forme d'un module donnant lieu à royalties, lorsqu'il s'agit de l'utiliser comme un module dans le cadre d'un développement ou d'une intégration propriétaire. En un mot, ils sont alors dans la position de vendre un composant c'est-à-dire non plus un bundle mais un élément d'un bundle à des entreprises dont le modèle économique consiste précisément à assurer la vente de bundles. Et ce composant est véritablement sur étagère («off the shelf») dans la mesure où son design n'est pas défini par un donneur d'ordre : il correspond à une ou plusieurs fonctionnalités qui peuvent être intégrées dans un nombre varié d'applications et de solutions grâce à nouveau à l'adaptabilité des biens logiciels lorsqu'ils sont disponibles sous la forme de codes sources. Dans un monde d'intégrateurs, ces quasi-éditeurs peuvent parvenir à s'assurer d'un flux de revenu substantiel, et de bénéficier de marges peut-être plus importantes que des intégrateurs spécialisés appelés à devenir des sous-traitants pour des intégrateurs plus généralistes et à travailler «à façon». Pour autant, de nombreux autres critères interviendront sans doute dans le choix d'un modèle d'intégrateurs spécialisé ou de quasi-éditeur, comme le cash-flow, et plus généralement les risques, mais aussi l'histoire particulière de chaque entreprise. Ces deux modèles économiques peuvent d'ailleurs se recouper, puisqu'il est vraisemblable que les quasi-éditeurs soient amenés à proposer des services d'intégrateurs spécialisés[10].

Les intégrateurs de systèmes informatiques seront-ils alors de «véritables» intégrateurs, tournés vers les services et intégrant du matériel, ou bien des OEM, c'est-à-dire des constructeurs qui feraient usage de composants logiciels adaptés à leurs plates-formes matérielles ? Un critère tient sans doute à la variété des besoins des clients. Si ceux-ci peuvent s'organiser autour d'architectures en nombre relativement limitées, et où la dimension matérielle serait importante, alors les OEM auront vraisemblablement un rôle prééminent. Ils pourraient dès lors, comme les constructeurs automobiles, avoir un rôle plus important dans la définition et le design des composants, bien entendu matériels, mais aussi logiciels, qui pourraient leur être fournis par des quasi-éditeurs ; ils feraient aussi dans une certaine mesure appel à des intégrateurs spécialisés en sous-traitance. A contrario, et dans l'hypothèse où les architectures seraient véritablement variables et devraient être très souvent adaptées aux besoins des clients, le marché pourrait favoriser les intégrateurs orientés vers les services, qui feraient alors largement appel à des intégrateurs spécialisés en sous-traitance et utiliseraient des composants véritablement sur étagère, puisque moins spécifiques. L'adaptabilité des bien logiciels rend ce second modèle moins coûteux qu'il n'y paraît, et augmente donc ses chances de s'imposer, bien que l'on puisse vraisemblablement imaginer une coexistence et une compétition entre ces deux formes d'intégrateurs.

12.5 Conclusion : vers une nouvelle économie du logiciel ?

Quoi qu'il en soit, là est peut-être tout simplement la nouvelle économie du logiciel : l'ouverture des sources rend accessible le modèle modulaire et développe donc le rôle de l'intégration. Cette nouvelle économie du logiciel s'accompagne donc de l'apparition de nouveaux acteurs économiques ainsi que de l'évolution d'acteurs existants vers des modèles économiques d'intégrateurs plus ou moins spécialisés, mais aussi de l'apparition de quasi-éditeurs de composants logiciels open-source. Ces évolutions fondent donc une nouvelle organisation industrielle du secteur de l'informatique : une nouvelle économie du logiciel tournée vers les clients, de plus en plus adaptée à leurs demandes particulières, grâce à l'intervention d'intégrateurs et d'intégrateurs spécialisés servis par des éditeurs de progiciels et de composants logiciels mais aussi par des quasi-éditeurs de composants logiciels ouverts : cf. Figure 2.

En ce sens, ce qui entravait la mise en place d'un modèle de composants, c'était aussi l'organisation du

[10]Cette question se pose sans doute de manière particulièrement aiguë pour les entreprises qui s'orientent vers des stratégies de «libération» d'un logiciel (voir l'article de François Horn dans cet ouvrage).

Figure 12.2 **—** La nouvelle économie du logiciel.

Clients

Nouveaux Intégrateurs de Systèmes
OEM (constructeurs) vs. SSII

Intégrateurs spécialisés, SS2L

Quasi-éditeurs de composants open-source
Fournisseurs de composants propriétaires

Editeurs de logiciels spécifiques et de progiciels

secteur de l'informatique qui n'y était pas adaptée : l'irruption des logiciels ouverts semble susceptible d'entraîner des mutations suffisamment profondes pour susciter les changements et les évolutions nécessaires à la mise en place progressive d'un véritable modèle de composants logiciels, y compris au sens d'une organisation industrielle renouvelée.

Au sein de cette nouvelle économie du logiciel, l'absence de «manufacturiers» est alors un peu une illusion (Hippel [von]). Cette illusion est due aux caractéristiques particulières des biens logiciels et à l'absence, en ce qui les concerne, de ligne de fabrication au sens strict, mais aussi au fait qu'ils se conçoivent dans une économie de bundles où le hardware, mais aussi et surtout les services, jouent un rôle essentiel. Au contraire, l'apparition des logiciels libres, le développement des composants et des intégrateurs, irait plutôt dans le sens d'un renforcement des processus manufacturiers dans le secteur de l'informatique et du logiciel, beaucoup plus intégrés d'une certaine manière, et beaucoup plus organisés même s'il n'ont plus grand chose à voir avec une manufacture d'épingles. Si les logiciels libres permettent à des utilisateurs avertis d'utiliser de se passer alors de fournisseur - de bénéficier directement du travail d'autres utilisateurs - il reste qu'un grand nombre, pour ne pas dire le plus grand nombre des clients professionnels optera, dans le domaine de l'informatique, pour une désintégration des aspects les moins «spécifiques» (Williamson [1985]). En un mot, il ne faut pas confondre la question de l'organisation industrielle et la question de l'innovation. Ces deux questions sont complémentaires mais ne se réduisent pas l'une à l'autre. Si la question de l'innovation dans le monde du logiciel est bien la question de ses «sources» (Hippel [von], Dalle [2002]) -c'est dans ce cadre que le rôle important des usagers apparaît clairement- il reste que ce sont les intégrateurs de systèmes (Pavitt [2002]), spécialisés ou non, qui répondent aux besoins des usagers, et qui auront pour tâche d'assurer la *manufacture* d'une solution informatique adaptée à chaque demande, à partir de composants logiciels, libres et/ou propriétaires.

Force est alors de constater que l'apparition des logiciels libres n'a pas fait du logiciel une commodité qui pourrait être distribuée comme une matière première, contrairement à ce qui a un temps été pensé : la nature même des biens logiciels, et notamment leur adaptabilité et la nécessité de leur adaptation à des besoins très hétérogènes, s'oppose à cette vision un peu réductrice, et conduit vers une organisation industrielle plus riche, et tout à fait différente, où le travail de transformation et d'adaptation est généralement conduit à façon par des intégrateurs plutôt que par les usagers eux-mêmes.

Bibliographie

G. AHARONIAN, 2000. Patent examination system is intellectually corrupt. URL http://www.bustpatents.com/corrupt.htm.

. AMBERLAB, Mose. .... URL http://amberlab.net/ubp/.

W. B. ARTHUR, 1989a. Competing technologies, increasing returns and lock-in by historical events : The dynamics of allocations under increasing returns to scale. *Economic Journal*, 99 :116–131.

C. G. AU PLAN, 2002. Économie du logiciel : renforcer la dynamique fraçaise. rapport du groupe présidé par Hugues Rougier, Rapporteur Général : Jean-Michel Dalle, Rapporteur : Sylvie Bénard.

A. BERTRAND-DOULAT, 1997. Les dessins et modèles. In M. VIVANT, éditeur, *Les créations immatérielles et le droit*. Ellipses.

J. BESSEN, 2002. Open source software : Free provision of complex public goods. Research on Innovation. URL http://www.researchoninnovation.org/online.htm#oss.

C. BESSY et E. BROUSSEAU, décembre 1998. Technology licencing contracts : features and diversity. *International Review of Law and Economics*, 18 :451–489.

C. BESSY et E. BROUSSEAU, 2001. Contrats de licence et innovation. In P. MUSTAR et H. PENAN, éditeurs, *Encyclopédie de l'innovation*. Economica, Paris.

A. BEUGNARD, 2001. Nouvelle économie du logiciel... un point de vue. projet «Nouvelle économie du Logiciel», RNTL. URL http://www-eco.enst-bretagne.fr/Etudes_projets/RNTL/workshop1/Beugnard.%pdf, 1re réunion de travail.

A. BOUSQUET et M. WOLKOWICZ, 1 ^{e} trim. 1994. Les enjeux économiques dans les contrats de licence. *L'écho des recherches*, 155.

R. BOYER et M. FREYSSENET, 1995. Émergence de nouveaux modèles industriels. hypothèses et démarches d'analyse. Technical Report 15, actes du GREPISA.

R. BOYER et M. FREYSSENET, 2000. *Les modèles productifs*. Collection Repères, La Découverte, Paris.

P. BRETON, 1990. *Une histoire de l'informatique*. Point Sciences, Le Seuil, Paris.

. CARDIONEWS, Mose. Site de l'association de cardiologues cardionews. URL http://www.cardionews.com/.

M. CASTELLS, 1998. *La Société en Réseaux*. Fayard.

M. CLÉMENT-FONTAINE, 1999. Étude de la gnu gpl. Mémoire de DEA, Université de Droit, Montpellier. http://crao.net/gpl/gpl.html.

W. M. COHEN et D. A. LEVINTHAL, 1989. Innovation and learning : The two faces of r&d. *Economic Journal*, 99 :569–596.

P. COHENDET, F. CREPLET, et O. DUPOUET, 2002. Innovation organisationnelle, communautés de pratique et communautés épistémiques : le cas de linux. *Revue française de gestion*, à paraître.

P. COHENDET, F. CRÉPLET, et O. DUPOUET, 2001. Interactions between epistemic communities and communities of practice as a mechanism of creation and diffusion of knowledge. In J.-B. ZIMMERMANN et A. KIRMAN, éditeurs, *Interaction and Market Structure*. Springer, Londres.

C. COLOMBET, 1999. *Propriété littéraire et artistique et droits voisins*. Précis Dalloz.

G. CORNU, 1987. *Vocabulaire juridique*. Association Henri Capitant, P.U.F.

J.-M. DALLE, Juillet 1995. Dynamiques d'adoption, coordination et diversité : la diffusion des standards technologiques. *Revue économique*, 46(4) :1081–1098.

J.-M. DALLE, 2002. Open-code : The sources of open-source innovations. In L. DAVIS, éditeur, *The Changing Role of Intellectual Property Rights*. Economics of Innovation and New Technologies. à paraître.

J.-M. DALLE et N. JULLIEN, 2000. Nt vs. linux, or some explorations into the economics of free software. In G. BALLOT et G. WEISBUCH, éditeurs, *Applications of Simulation to Social Sciences*, pages 399–416. Hermès, Paris.

J.-M. DALLE et N. JULLIEN, janvier 2003. 'libre' software : Turning fads into institutions ? *Research Policy*, 32(1) :1–11.

P. A. DAVID, mai 1985. Clio and the economics of qwerty. *American Economic Review*, 75-2 Papers and Proceeding :332–337.

P. A. DAVID et D. FORAY, 1995. Dépendance du sentier et économie de l'innovation : un rapide tour d'horizon. In L. BENZONI, M. MOUGEOT, et A. TORRE, éditeurs, *Économie industrielle : développement récents*. Revue d'économie industrielle.

J. DE BANDT, 1995. *Services aux entreprises : informations, produits, richesses*. Economica, Paris.

P. B. DE LAAT, 2000. Patenting mathematical algorithms : What's the harm. *International Review of Law and Economics*, 20.

M. DELAPIERRE, L.-A. GERARD-VARET, et J.-B. ZIMMERMANN, décembre 1980. Choix publics et normalisation des réseaux informatiques. Technical report, Rapport BNI.

M. DELAPIERRE et J.-B. ZIMMERMANN, 1994. La nature du produit informatique. *Terminal*, 65, automne : 87–104.

G. DRÉAN, 1996. *L'industrie informatique, structure, économie, perspectives*. Masson, Paris.

Y. DUPUIS et O. TARDIEU, 2001. Les brevets superflus en matière de logiciel. rapport, École des Mines, Paris.

. EUROSTAF, 2000. L'informatique : le passage d'une logique de produit à une logique de services. Technical report, Eurostaf.

J. FARRELL, printemps 1989. Standardizationand intellectual property. *Jurimetrics Journal*.

J. FARRELL et G. SALONER, 1988. *The Economics of Converters*. MIT.

B. FAUCON et J.-P. SMETS-SOLANES, 1999. *Logiciels libres. Liberté, égalité, business*. Éditions Edispher, Paris.

D. FORAY, 2e trimestre 1989. Les modèles de compétition technologiques, une revue de la littérature. *Revue d'économie industrielle*, 48 :16–34.

D. FORAY, 1er trimestre 1990. Exploitation des externalités de réseau versus évolution des normes : les formes d'organisation face au dilemme de l'efficacité, dans le domaine des technologies de réseau. *Revue d'économie industrielle*, 51 :113–140.

D. FORAY, 2002. Innovation et concurrence dans les industries de réseau. *Revue Fraçaise de Gestion*, pages 131–154.

D. FORAY et J.-B. ZIMMERMANN, octobre 2001. L'économie du logiciel libre : organisation coopérative et incitation à l'innovation. *Revue économique*, 52 :77–93. Hors série, numéro spécial : économie d'Internet, sous la direction d'É. Brousseau et N. Curien.

J. FOYER et M. VIVANT, 1991. *Le droit des brevets*. Thémis, PUF, Paris.

J. GADRAY, 1998. La caractérisation des biens et des services, d'adam smith à peter hill : une approche alternative. Technical report, IFRESI, Lille. document de travail.

N. T. GALLINI et R. A. WINTER, 1985. Licensing in the theory of innovation. *RAND Journal of Economics*, 16(2).

R. GARUD et A. KUMARASWANY, 1993. Changing competitive dynamics in network industries : An exploration of sun microsystems' open systems strategy. *Strategic Management Journal*, 14 :351–369.

C. GENTHON, 1995. *Croissance et crise de l'industrie informatique mondiale*. Syros, Paris.

C. GENTHON, 2000. Le cas sun microsystem. ENST Bretagne. URL http://www-eco.enst-bretagne.fr/Enseignement/2A/1999-2000/EST201/sun/su%n00.htm, support de cours.

C. GENTHON, 2001. Le libre et l'industrie des services et logiciels informatique. projet «Nouvelle économie du Logiciel», RNTL. URL http://www-eco.enst-bretagne.fr/Etudes_projets/RNTL/workshop1/genthon.p%df, 1re réunion de travail.

J.-Y. GOFFI, 1996. *La philosophie de la technique*. Que sais-je ?, PUF, Paris.

L.-A. GÉRARD-VARET et J.-B. ZIMMERMANN, mai 1985. Concept de produit informatique et comportement des agents de l'industrie. In *colloque «Structures économiques et économétrie»*, mai 1985.

. GUIDE LAMY, 2000. *Droit de l'Informatique*, volume 3415. Guide Lamy.

P. HIMANEN, 2001. A brief history of computer hackerism. URL http://www.hackerethic.org/writtings/hackerhistory.shtml.

E. HIPPEL (VON), Juillet 1986. Lead users : a source of novel product concepts. *Management Science*, 32(7).

E. HIPPEL (VON), 1988. *The Sources of Innovation*. Oxford University Press, New York.

E. HIPPEL (VON), 2001. Open-source shows the way : Innovation by and for users - no manufacturer required ! *Sloan Management Review*, été.

F. HORN, 1999. L'importance du logiciel libre dans l'amélioration de l'efficience des logiciels. *Terminal*, 80/81 :119–148. Numéro spécial : le logiciel libre.

F. HORN, 2000b. *L'économie du logiciel. Tome 1 : De l'économie de l'informatique à l'économie du logiciel. Tome 2 : De l'économie du logiciel à la socio-économie des «mondes de production» des logiciels*. Thèse de doctorat, Université de Lille I, mention : économie industrielle, 570 pages. URL http://www-eco.enst-bretagne.fr/Etudes_projets/RNTL/documents_universit%aires.html.

N. JULLIEN, 1999. Linux : la convergence du monde unix et du monde pc. *Terminal*, 80/81 :43–70. Numéro spécial : le logiciel libre.

N. JULLIEN, novembre 2001. *Impact du logiciel libre sur l'industrie informatique*. Thèse de doctorat, Université de Bretagne Occidentale / ENST Bretagne, mention : sciences économiques, 307 pages. URL http://www-eco.enst-bretagne.fr/Etudes_projets/RNTL/documents_universit%aires.html.

N. JULLIEN et J.-B. ZIMMERMANN, 2002. Le logiciel libre : une nouvelle approche de la propriété intellectuelle ? *Revue d'économie industrielle*, 99 :159–178. numéro spécial : les droits de propriété intellectuelle : nouveaux domaines, nouveaux enjeux.

M. L. KATZ et C. SHAPIRO, août 1986. How to license intangible property. *The Quaterly Journal of Economics*, 406 :567–589.

M. KOTABE, A. SAHAY, et P. S. AULAKH, 1996. Emerging role of technology licensing in the development of global product strategy : conceptual framework and research propositions. *Journal of Marketing*, 60 : 73–88.

J.-J. LAFFONT, 1991. *Fondements de l'économie publique*. Economica, Paris.

K. LAKHANI et E. VON HIPPEL, 2000. How open source software works : Free user to user assistance. URL http://opensource.mit.edu/online-pap-abst.html, travail de recherche.

R. N. LANGLOIS, 2001. The vanishing hand : the modular revolution in american business. mimeo.

J. LERNER et J. TIROLE, 2002. Some simple economics of open source. *Journal of Industrial Economics*, 52. URL http://opensource.mit.edu/online-pap-abst.html. à paraître.

. LES ÉTUDES DU CONSEIL D'ÉTAT, 1998. *Internet et les réseaux numériques*. La Documentation Française, Paris.

X. LINANT DE BELLEFONDS, janvier 2001. Brevetabilité du logiciel ? à revoir. *Communication - commerce électronique*.

janvier 2002a. *IBM-Linux : une réalité économique*, Paris. Linux Expo 2002. conférence.

janvier 2002b. *The Inevitable Success of the Open Source Technology Model*. Linux Expo 2002, Paris. Conférence.

janvier 2002c. *Un business model gagnant pour un éditeur de logiciel libre*. Linux Expo 2002, Paris. conférence.

A. LUCAS et H.-J. LUCAS, 2001. *Traité de la propriété littéraire et artistique.*, volume 419. Litec (2^d ed.).

. MANDRAKESOFT/BOURSORAMA, 2002. Suivre le cours en bourse de MandrakeSoft. URL http://www.boursorama.com/cours33.phtml?symbole=1r04477-OTC.

J. MATEOS GARCIA, 2001. Innovating without money. Linux and the Open Source paradigm as an alternative to commercial development. Msc Dissertation, SPRU, University of Sussex.

. MATHÉLY, 1991. *Le nouveau droit français des brevets d'invention*. Librairie du journal des Notaires et des Avocats, Paris.

D. MAUME, 7 janvier 2002. Linux s'impose dans les entreprises. *L'Usine Nouvelle*.

MOSE, 2002. Site de Makina Corpus dédié à la Communauté. URL http://www.makina-corpus.org/.

A. MOULINE, 1996. *L'industrie des Services Informatiques*. Economica, Collection Économie Poche, Paris.

D. C. MOWERY, éditeur, 1996. *The International Computer Software Industry, A comparative Study of Industry Evolution and Structure*. Oxford University Press.

. ORBITEN, avril 2000. Free software survey. URL http://orbiten.org/ofss/01.html.

K. PAVITT, juin 2002. System integrators as «post-industrial» firms ? In *Industrial Dynamics of the New and Old Economy - who embraces whom ?* DRUID Summer Conference, juin 2002.

F. POLLAUD-DULLIAN, 1999. *Droit de la propriété industrielle*, volume 238. Montchrestien, Paris.

A. R., 4 Juillet 2001. Pour les ingénieurs d'alcôve, le temps aussi est libre. *Les Echos*.

E. S. RAYMOND, 1998a. The Cathedral and the Bazaar. URL http://www.tuxedo.org/~esr/writings/cathedral-bazaar/.

E. S. RAYMOND, 1998b. Homesteading the Noosphere. URL http://www.tuxedo.org/~esr/writings/homesteading/.

E. S. RAYMOND, 1999. *The Cathedral & the Bazaar ; Musing on Linux and Open Source by Accidental Revolutionary*. O'Reilly, Sebastopol, Calif.

G. B. RICHARDSON, avril 1997. Economic analysis, public policy and the software industry. In *The Economics of Imperfect Knowledge - Collected papers of G.B. Richardson*, volume 97-4. Edward Elgar, DRUID Working Paper.

D. K. ROSENBERG, 2000. *Open Source. The unauthorized white papers*. IDG Books Worldwide, inc.

. ROUBIER, 1952. *Le droit de la propriété industrielle, Tome 1*. Recueil Sirey, Toulouse.

. ROUBIER, 1954. *Droit de la propriété industrielle*. Recueil Sirey, Paris.

C. SHAPIRO et H. VARIAN, 1999. *Information Rules : A Strategic Guide to the Network Economy*. Harvard Business School Press.

C. SHAPIRO et H. VARIAN, 2000. *Économie de l'information - guide stratégique de l'économie de réseau*. De Boeck Université. Traduction française de Shapiro et Varian [1999].

P. SIRINELLI, 1992. *Propriété littéraire et artistique et droits voisins*. Mémentos Dalloz.

J.-M. THIÉRY, 1999. Quatre décennies de logiciels et de données scientifiques libres. *Terminal*, 80/81 : 191–202. Numéro spécial : le logiciel libre.

M. VIVANT, éditeur, 1997. *Les créations immatérielles et le droit*. Ellipses, Paris.

M. VIVANT, Juillet 2002. intervention. In *Propriété intellectuelle et mondialisation*. ERCIM, à paraître, Juillet 2002.

M. VIVANT et . AL., 2001. *Droit de l'Informatique*, volume 192. Guides Lamy.

M. VIVANT et C. LE STANC, 2002. *Droit de l'Informatique*, volume 151. Guides Lamy.

F. VON GOTTL-OTTLILIENFELD, 1914. *Précis d'économie sociale*, chapter Économie et technique. .

M. WEBER, 1971. *Économie et société*. Plon, Paris.

D. A. WHEELER, 2001. Why Open Source Software/Free Software (OSS/FS) ? Look at the Numbers. URL http://www.dwheeler.com/.

O. E. WILLIAMSON, 1985. *The Economic Institutions of Capitalism*. Free Press, New-York.

O. E. WILLIAMSON, 1996. *The Mechanisms of Governance*. Oxford University Press, New York.

J.-B. ZIMMERMANN, 1er trimestre 1989. Groupes industriels et grappes technologiques. *Revue d'économie industrielle*, 47(5).

J.-B. ZIMMERMANN, septembre 1995a. Le concept de grappes technologiques. un cadre formel. *Revue économique*, 46(5) :1263–1295.

J.-B. ZIMMERMANN, 1995b. L'industrie du logiciel : de la protection à la normalisation. In M. BASLE, D. DUFOURT, J.-A. HÉRAUD, et J. PERRIN, éditeurs, *Changement institutionnel et changement technologique. Évaluation, droits de propriété intellectuelle, système national d'innovation*, pages 195–207. CNRS Éditions, Paris.

J.-B. ZIMMERMANN, 1998. Un régime de droit d'auteur : la propriété intellectuelle du logiciel. *Réseaux*, 88-89 :91–106.

J.-B. ZIMMERMANN, 1999. Logiciel et propriété intellectuelle : du copyright au copyleft. *Terminal*, 80/81 : 95–116. Numéro spécial : le logiciel libre.